**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΤΩΝ ΚΑΤΕΡΓΑΣΙΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

# Μελέτη με προσομοίωση της πλοήγησης σε εσωτερικό χώρο ενός drone εξυπηρέτησης ευέλικτου συστήματος κατεργασιών

ΒΛΑΧΟΠΑΝΟΥ ΣΟΦΙΑ

Επιβλέπων: Γ.Χ Βοσνιάκος, Καθηγητής Ε.Μ.Π

Αθήνα, Ιούνιος 2024

*<u>Υπεύθυνη δήλωση</u> για λογοκλοπή και για κλοπή πνευματικής ιδιοκτησίας:*

**Έχω διαβάσει και κατανοήσει τους κανόνες για τη λογοκλοπή και τον τρόπο σωστής αναφοράς των πηγών που περιέχονται στον οδηγό συγγραφής Διπλωματικών Εργασιών. Δηλώνω ότι, από όσα γνωρίζω, το περιεχόμενο της παρούσας Διπλωματικής Εργασίας είναι προϊόν δικής μου εργασίας και υπάρχουν αναφορές σε όλες τις πηγές που χρησιμοποίησα.**

**Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτή τη Διπλωματική εργασία είναι του συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις της Σχολής Μηχανολόγων Μηχανικών ή του Εθνικού Μετσόβιου Πολυτεχνείου.**

**ΒΛΑΧΟΠΑΝΟΥ ΣΟΦΙΑ**

# ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία μου δίνει την δυνατότητα να εκφράσω την ευγνωμοσύνη μου σε ορισμένους ανθρώπους οι οποίοι με βοήθησαν τόσο στην διεκπεραίωση της όσο και στην ολοκλήρωση της πορείας μου ως φοιτήτρια του τμήματος Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου.

Πρωτίστως, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου , κύριο Γ.Χ. Βοσνιάκο, καθηγητή της Σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου. Ο κύριος Βοσνιάκος ήταν συνεχώς δίπλα μου και με στήριζε σε όλη την πορεία της εργασίας τόσο με τις χρήσιμες καθοδηγήσεις του όσο και με την άμεση ανταπόκριση του σε οποιαδήποτε έκκλησή μου. Την ίδια στιγμή, θα ήθελα να τον ευχαριστήσω και για την ανάθεση του παρόντος θέματος καθώς πρόκειται για ένα αντικείμενο που θα με συνοδεύει στο μελλοντικό μου ακαδημαϊκό ταξίδι.

Έπειτα, θα ήθελα να εκφράσω τις ευχαριστίες μου στους γονείς και στην αδερφή μου οι οποίοι ήταν στο πλευρό μου σε κάθε βήμα και επιλογή μου κατά την διάρκεια της φοίτησής μου.

Τέλος, επιθυμώ να εκφράσω τις ειλικρινείς και θερμές μου ευχαριστίες προς όλους τους φίλους μου, οι οποίοι με στηρίζουν αδιάλειπτα με κάθε δυνατό τρόπο στις δύσκολες στιγμές και μοιράζονται μαζί μου τη χαρά στις ευχάριστες περιστάσεις

# ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διπλωματικής εργασίας αποτελεί η προσομοίωσης της πτήσης σε εσωτερικό χώρο ενός drone το οποίο είναι κατασκευασμένο για την εξυπηρέτηση ευέλικτων συστημάτων κατεργασιών. Η προσομοίωση αυτή έγινε με χρήση του ROS(Robot Operating System) στο περιβάλλον της εφαρμογής Gazebo. Ο σχεδιασμός των αλγορίθμων χρησιμοποίησε την γλώσσα προγραμματισμού C++, η οποία είναι συνηθισμένη στα ρομποτικά συστήματα. Παράλληλα, στα πλαίσια της ίδιας εργασίας παρουσιάζεται ο σχεδιασμός και η κατασκευή των ποδιών που ενσωματώνονται στο drone του εργαστηρίου του τομέα Τεχνολογίας των Κατεργασιών της Σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου.

Ξεκινώντας με την δημιουργία των ποδιών, η διπλωματική εργασία παρουσιάζει την μετάβαση από διάφορες μορφές και ιδέες. Αναλυτικότερα, περιγράφεται, στην αρχή, η αλλαγή του σκελετού του κατασκευασμένου μη επανδρωμένου εναέριου οχήματος. Αυτός μεταβάλλει αφενός το υλικό του και αφετέρου τις διαστάσεις και τις κατεργασίες που οδηγούν στην υλοποίησή του. Παράλληλα, εξαιτίας πρακτικών λόγων τα πόδια του αεροσκάφους αλλάζουν τρόπο σύνδεσης με το υπόλοιπο σώμα με αποτέλεσμα να υπάρχει μεταβολή στην δομή τους. Ολοκληρώνοντας, καθοριστικές αλλαγές συντελούνται και στο σχήμα και το υλικό κατασκευής καθώς είναι επιθυμητή η δοκιμή άλλης λογικής και η κατεργασία της φρέζας απομακρύνεται από το προσκήνιο.

Επικεντρώνοντας το ενδιαφέρον στο ζήτημα της προσομοίωσης, στόχος της εργασίας αποτελεί η ενσωμάτωση των απαραίτητων αισθητήρων που θα εξασφαλίζουν ομαλή πτήση και πλοήγηση σε εσωτερικό χώρο. Αυτοί επιλέχθηκαν να είναι το γυροσκόπιο (IMU) και ο αισθητήρας Ultra wide band(UWB). Ο πρώτος είναι κοινός για τα μη επανδρωμένα εναέρια οχήματα με αποτέλεσμα να προστίθεται με την μορφή plugin στο σχεδιασμένο αλγόριθμο. Ο δεύτερος αποτελεί καινοτόμο σύστημα το οποίο βρίσκει εφαρμογή σε εσωτερικούς χώρους. Αυτό σημαίνει ότι ο κώδικας που περιγράφει την λειτουργία του δεν μπορεί να βρεθεί σε μορφή plugin και γράφτηκε από την αρχή με σκοπό να καλύψει τις απαιτήσεις της διπλωματικής εργασίας. Η λειτουργία του συστήματος και του κώδικα διασφαλίζεται με την δημιουργία πέντε σεναρίων πτήσης του οχήματος. Αυτά παρουσιάζονται αναλυτικά σε μορφή διαγραμμάτων και εξηγούνται κατάλληλα.

# ABSTRACT

The goal of this thesis is to simulate the indoor flight of a drone designed to serve flexible machining systems. This simulation was run in the Gazebo application environment with ROS (Robot Operating System). The algorithm was designed using the C++ programming language, which is widely used in robotic systems. Simultaneously, the design and construction of the legs integrated into the drone of the laboratory of the Department of Machining Technology of the Faculty of Mechanical Engineering of the National Technical University of Athens are presented.

Beginning with the creation of the legs, the thesis depicts the progression through various forms and ideas. In more detail, the change in the frame of the constructed unmanned aerial vehicle is described first. He alters the material, as well as the dimensions and machining processes that lead to its realization. At the same time, for practical reasons, the aircraft's legs change how they connect to the rest of the body, resulting in a structural change. Finally, significant changes occur in the shape and material of the construction, as it is desirable to test another logic and the milling machine is removed from the foreground.

Focusing on simulation, the project's goal is to integrate the necessary sensors to ensure smooth flight and navigation in an indoor environment. The gyroscope (IMU) and Ultra wide band (UWB) sensors were selected. The former is common for unmanned aerial vehicles, so it is added as a plugin to the designed algorithm. The second is an innovative system with indoor applications. This means that the code describing its operation cannot be found as a plugin and had to be written from scratch to meet the thesis requirements. Creating five vehicle flight scenarios ensures that the system and code function properly. These are presented in detail as diagrams and appropriately explained.

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

# ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

| | |
|---|---|
| U.A.V | Unmanned Aerial Vehicle |
| P.C.B.A | Printed Circuit Board Assembly |
| P.C.B | Printed Circuit Board |
| S.O.C | System on a Chip |
| Η.Π.Α | Ηνωμένες Πολιτείες Αμερικής |
| U.S.A.F | United States Air Force |
| G.N.S.S | Global Navigation Satellite System |
| G.P.S | Global Positioning System |
| U.W.B | Ultra Wide Band |
| R.O.S | Robotic Operating System |
| C.N.C | Computer Numerical Control |
| P.L.A | Polylactic Acid |
| X.M.L-R.P.C | Extensible Markup Language – Remote Procedure Call |
| P.I.D | Proportional Integral Derivative |
| I.M.U | Inertial Measurement Unit |
| F.C.C | Federal Communications Commission |
| T.o.F | Time of Flight |
| T.D.o.A | Time Difference of Arrival |
| P.D.o.A | Phase Difference of Arrival |

# 1 ΕΙΣΑΓΩΓΗ

## 1.1 Γενικά για τα drones

Ένα μη επανδρωμένο εναέριο όχημα, κοινώς αναφερόμενο ως drone, αποτελεί ένα αεροσκάφος χωρίς ανθρώπινο παράγοντα τόσο στον πιλότο όσο και στο πλήρωμα ή τους επιβάτες. Ο έλεγχος της πτήσης του πραγματοποιείται απομακρυσμένα με την χρήση τηλεχειριστηρίου από έναν χειριστή. Παράλληλα, μπορεί να προγραμματιστεί η αυτόματη μετακίνησή του υπακούοντας σε προκαθορισμένες πορείες και εντολές. Αυτό δύναται να υλοποιηθεί με διάφορους βαθμούς αυτονομίας όπως με τη βοήθεια αυτόματου πιλότου, έως και την δημιουργία ενός πλήρως αυτόνομου αεροσκάφους που δεν έχει καμία πρόβλεψη για ανθρώπινη παρέμβαση. [1]

Η δομή ενός τέτοιου αεροσκάφους απαρτίζεται από το υλικό(Hardware) , το λογισμικό (Software) και το μηχανολογικό (Mechanical) σύστημα. Ξεκινώντας με το πρώτο, το βασικό του χαρακτηριστικό αποτελεί ένα τυπωμένο συναρμολόγημα πλακέτας κυκλώματος(PCBA) και ένα πολυστρωματικό PCB που φιλοξενεί το SOC (System On a Chip) καθώς και διάφορα εξαρτήματα των υποσυστημάτων που συνδέονται μεταξύ τους είτε μέσω χάλκινων ιχνών είτε με καλώδια. Στην Εικόνα 1.1-1Εικόνα 1.1-1 παρατηρείται η πλακέτα συναρμολογημένη με το SOC και τα υποσυστήματα στην κορυφή. Πιο συγκεκριμένα, το SOC είναι ένας μικροσκοπικός υπολογιστής σε μορφή τσιπ ο οποίος περιγράφεται από μια συσκευή ημιαγωγών και ένα ολοκληρωμένο κύκλωμα το οποίο ενσωματώνει ψηφιακές, αναλογικές, μικτού σήματος και ραδιοσυχνοτήτων συσκευές. Την ίδια στιγμή, το υλικό τμήμα περιλαμβάνει και επιμέρους υποσυστήματα όπως συσκευές για είσοδο, έξοδο, αποθήκευση και πλοήγηση. [1]


**Εικόνα 1.1-1: Πλήρως συναρμολογημένη πλακέτα [1]**

Ταυτόχρονα, το ζήτημα του λογισμικού συστήματος που πρέπει να χρησιμοποιηθεί σε drone δύναται να διαχωριστεί σε τέσσερις κατηγορίες. Αυτές είναι τα συστατικά Firmware, το λειτουργικό σύστημα και τα προγράμματα οδήγησης, ο έλεγχος και οι εφαρμογές. Ωστόσο, την ταυτότητα του εναέριου οχήματος καθορίζει το μηχανολογικό του σύστημα. Αυτό αποτελείται από τα περιβλήματα, τον παράγοντα μορφής και τον μηχανολογικό σχεδιασμό. Συνοπτικά, αυτό περιλαμβάνει μια πληθώρα μηχανικών μερών συνδεδεμένα με περίπλοκο τρόπο με ηλεκτρικά τμήματα μέσω μηχανικών ή και θερμικών διασυνδέσεων. [1]

Η πιο δημοφιλής μορφή μη επανδρωμένου αεροσκάφους είναι αυτή του τετρακόπτετρου (quadcopter) κατασκευασμένο με ένα πλαίσιο σχήματος X ή H. Αυτό, συνήθως, περιλαμβάνει μονάδες σερβοκινητήρων σε κάθε άκρο, το PCBA και τον ελεγχτή κλεισμένα σε πλαστικό κάλυμμα ,έλικες, σύστημα προσγείωσης και μια μπαταρία. Ένα drone με την περιγραφόμενη μορφή και σε σχήμα X παρουσιάζεται στην Εικόνα 1.1-2.

**Εικόνα 1.1-2: Τετρακόπτερο σε δομή σχήματος Χ**

## 1.2 Ιστορική αναδρομή

Η ανάπτυξη των drones ξεκίνησε στις αρχές του 20ού αιώνα με την μορφή της εκπαίδευσης του στρατιωτικού προσωπικού ως στόχοι. Το 1883, ο Douglas Archibald χρησιμοποίησε ένα ανεμόμετρο δεμένο σε χαρταετό για να μετρήσει την ταχύτητα του ανέμου σε ύψος έως 1.200 μέτρα. Έπειτα, τρία χρόνια αργότερα τοποθέτησε κάμερες σε χαρταετούς, δημιουργώντας έτσι το πρώτο drone το οποίο έλαβε παγκόσμια αναγνώριση. Κατά την διάρκεια του πολέμου μεταξύ Ισπανών και Αμερικάνων, ο William Eddy χρησιμοποίησε τέτοιους χαρταετούς με σκοπό να τραβάει φωτογραφίες. Μέχρι τον Α' Παγκόσμιο Πόλεμο, τα drones είχαν ήδη αναγνωριστεί ως σημαντικά συστήματα. Ο Charles Kettering ανέπτυξε το πρώτο διπλής όψης αεροπλάνο για τον στρατό, γνωστό ως Kettering Aerial Torpedo ή "Bug Kettering". Αυτό το drone είχε πτερύγια που αποσπώνται και απελευθερώνονταν όταν έφτανε στον στόχο. Η λειτουργία του συνοψίζεται στην απόσταση των 40 μιλίων και την μεταφορά 180 λιβρών εκρηκτικών. Αυτό φαίνεται στην Εικόνα 1.2-1. [2]

**Εικόνα 1.2-1: Πρώτο αεροπλάνο διπλής όψης [2]**

Με την ολοκλήρωση του Α' Παγκόσμιου Πολέμου, η χρήση των drones και των συστημάτων τους έγινε εκτενέστερη. Αυτό οδήγησε στην δημιουργία εταιρειών αεροφωτογράφισης που χρησιμοποιούσαν αεροσκάφη για αποτυπώσεις και χαρτογράφηση. Το 1924, ο Archibald Montgomery Low έφτιαξε τον πρώτο

σύνδεσμο δεδομένων και επέλυσε ζητήματα παρεμβολών από τους κινητήρες των drones. Το γεγονός αυτό ακολούθησε η πρώτη επιτυχημένη πτήση με ραδιοεπικοινωνία από τον ίδιο. Το 1933, στη Μεγάλη Βρετανία, τρία τηλεχειριζόμενα αεροσκάφη, γνωστά ως "Fairey Queen", πέταξαν, αλλά τα δύο συνετρίβησαν. [2]

Κατά την Κρίση των Πυραύλων της Κούβας το 1962, αναπτύχθηκαν drones αναγνώρισης, γνωστά ως Fireflys, αλλά δεν χρησιμοποιήθηκαν λόγω της οικονομικής κρίσης της εποχής. Η Northrop-Ventura Division δημιούργησε μια επιτυχημένη σειρά drones στόχου, με όνομα Falconer, με ένα εξελισσόμενο σύστημα ραδιοελέγχου. Τον Αύγουστο του 1971, το Υπουργείο Άμυνας των ΗΠΑ ανέπτυξε μη επανδρωμένα συστήματα που είχαν ως σκοπό τον εντοπισμό πυροβολικών στόχων και καθορισμό λέιζερ, με ποσοστό επιτυχίας 90% μέχρι το τέλος του πολέμου του Βιετνάμ. [2]

Στη δεκαετία του 1990, τα drones εξελίχθηκαν και πέτυχαν μεγαλύτερη εμβέλεια και ακρίβεια. Κατά τον Πόλεμο του Κόλπου το 1991, χρησιμοποιήθηκαν τα drones με ονόματα Pioneer, ExDrone και Pointer από τις ΗΠΑ, Mini Avion de Reconnaissance Telepilot από τη Γαλλία και CL 89 από τη Βρετανία. Το 1995, το ΝΑΤΟ έστειλε drones στον πόλεμο της Βοσνίας για αναγνώριση και επιτήρηση αλλά οι θερμικές κάμερες που διέθεταν δεν μπορούσαν να διαπεράσουν τα σύννεφα. Αυτό οδήγησε στην δημιουργία ραντάρ συνθετικού διαφράγματος, τα οποία όμως ήταν βαριά με αποτέλεσμα την προσαρμογή του μεγέθους και της αντοχής των drones. Τη δεκαετία του 2000, δημιουργήθηκαν τα μοντέλα Predator B και C, για λήψεις σε μεγαλύτερα ύψη με χρήση αναβαθμισμένων συστημάτων. [2]

Το 2002, κατά τον πόλεμο του Ιράκ, τα drones αναδείχθηκαν ως βασικά οπλικά συστήματα. Επίσης, drones χρησιμοποιήθηκαν στο Αφγανιστάν αποδεικνύοντας τη στρατιωτική τους αξία. Την τελευταία δεκαετία, οι εξελίξεις στην τεχνολογία των ηλεκτρονικών βελτίωσαν την αποτελεσματικότητα και την αυτονομία των drones, καθιστώντας τα πιο προσιτά και παρέχοντας τους περισσότερες δυνατότητες. Το 2012, η USAF διέθετε 7.494 drones, και το 2013 χρησιμοποιήθηκαν σε πάνω από 50 χώρες, συμπεριλαμβανομένων της Κίνας, του Ιράν και του Ισραήλ και ορισμένες από αυτές ξεκίνησαν να δημιουργούν τις δικές τους τεχνολογίες. [1]

Οι ραγδαίες εξελίξεις της σύγχρονης εποχής στον τομέα των έξυπνων τεχνολογιών προκάλεσε ταυτόχρονη αύξηση της χρήσης μη επανδρωμένων αεροσκαφών για καταναλωτικές και γενικές αεροπορικές δραστηριότητες. Από το 2021, τα τετρακόπτερα drones αποτελούν παράδειγμα της ευρείας δημοτικότητας των ραδιοελεγχόμενων αεροσκαφών και παιχνιδιών για χόμπι. [1]

## 1.3 Drones σε εξωτερικό και εσωτερικό χώρο

Η επίτευξη της αυτόνομης συμπεριφοράς ενός drone αποτελεί σημαντική πρόκληση και απαιτεί την συνεργασία περίπλοκων υποσυστημάτων που βρίσκονται εντός του εναέριου οχήματος. Πιο αναλυτικά, η λειτουργία του στηρίζεται στην επαρκή απόδοση του συστήματος πλοήγησης και εντοπισμού του μη επανδρωμένου αεροσκάφους.

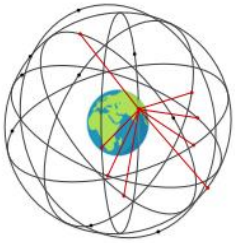
Ως εντοπισμός ή πλοήγηση ορίζεται η διαδικασία που ακολουθείται για την συγκέντρωση πληροφορίων που σχετίζονται με αντικείμενα που παρακολουθούνται και αφορούν πολλαπλά σημεία αναφοράς εντός μιας προκαθορισμένης περιοχής. Με άλλα λόγια, περιγράφεται ως μία προσπάθεια προσδιορισμού της θέσης τόσο κινητών όσο και σταθερών συσκευών με την χρήση σταθερών κόμβων. [3]

Ανάλογα με το περιβάλλον στο οποίο βρίσκονται και λειτουργούν τα drones χωρίζονται σε δύο μεγάλες κατηγορίες, αυτές του εσωτερικού και του εξωτερικού χώρου. Τα drones εξωτερικού χώρου είναι κατασκευασμένα για χρήση σε ανοικτά περιβάλλοντα με αποτέλεσμα ο σχεδιασμός τους να ανταποκρίνεται στην ανάγκη κάλυψης μεγάλων περιοχών με εκτεταμένο χρόνο πτήσης. Αυτός είναι ο λόγος που τα εναέρια οχήματα αυτής της κατηγορίας είναι μεγάλα, ανθεκτικά και εξοπλισμένα με προηγμένους αισθητήρες και κάμερες. [4]

Αντίθετα, τα drones εσωτερικού χώρου είναι μικρά, ελαφριά και πολύ ευέλικτα. Στόχος του σχεδιασμού αυτού είναι η πτήση σε χώρους με περιορισμούς και η γρήγορη εναλλαγή κατευθύνσεων χωρίς να θέτουν σε κίνδυνο τόσο τον εαυτό τους όσο και το περιβάλλον στο οποίο βρίσκονται και ενεργούν .Τα περισσότερα από τα drones εσωτερικού χώρου διαθέτουν, επίσης, αισθητήρες και κάμερες έτσι ώστε να τα βοηθούν στον εντοπισμό των εμποδίων και την αποφυγή συγκρούσεων. Τέλος, εξαιρετικά καθοριστικός για αυτή την κατηγορία εναέριων οχημάτων είναι ο έλεγχος του ύψους καθώς αυτά θα πρέπει να παραμένουν σταθερά σε συγκεκριμένη θέση καθώς επιχειρούν πορείες. [4]

Εκτός από τις διαφορές στην σχεδιαστική νοοτροπία οι δυο κατηγορίες drones παρουσιάζουν σοβαρές αντιθέσεις στην μέθοδο πλοήγησης. Όσον αφορά τα εναέρια οχήματα εξωτερικού χώρου, αυτά στηρίζονται στα σήματα του Παγκόσμιου Δορυφορικού Συστήματος Πλοήγησης (GNSS) το οποίο συμπεριλαμβάνει το Παγκόσμιο Σύστημα Εντοπισμού Θέσης (GPS), το Galileo και το Beidou. Το πρώτο από αυτά είναι το πιο ευρέως χρησιμοποιούμενο σύστημα δορυφορικού εξοπλισμού που παρέχει πληροφορίες γεωγραφικού εντοπισμού στους δέκτες του. [5]

Το Παγκόσμιο Σύστημα Εντοπισμού Θέσης (GPS) υπολογίζει την θέση του δέκτη αξιοποιώντας την ώρα και τη θέση του σήματος που προέρχεται από δορυφόρο. Ο κάθε ένας από τους δορυφόρους διαθέτει σταθερά ατομικά ρολόγια τα οποία συγχρονίζονται με τους υπόλοιπους καθώς και με τους επίγειους δέκτες. Ταυτόχρονα, εκπέμπει συνεχώς ένα ραδιοσήμα για να παρέχει την τρέχουσα κατάστασή του. Ο υπολογισμός της απόστασης μεταξύ ενός δορυφόρου και ενός δέκτη γίνεται με βάση τον χρόνο καθυστέρησης των μεταδιδόμενων και λαμβανόμενων σημάτων, καθώς τα ραδιοκύματα κινούνται με σταθερή ταχύτητα. Με άλλα λόγια, ένας δέκτης GPS υπολογίζει τις αποστάσεις από πολλαπλούς δορυφόρους και έτσι προσδιορίζει την ακριβή θέση του καθώς και την απόκλισή της από την πραγματική ώρα. Για να επιτευχθεί αυτό, πρέπει να χρησιμοποιηθούν τουλάχιστον τέσσερις δορυφόροι, οι οποίοι βοηθούν στον υπολογισμό των τεσσάρων άγνωστων παραμέτρων οι οποίοι απευθύνονται σε προσανατολισμό και χρόνο. Επιπλέον, το δορυφορικό σύστημα GPS περιλαμβάνει από 24 έως 32 δορυφόρους που βρίσκονται σε τροχιά γύρω από την γη, εξασφαλίζοντας ακρίβεια αποτελεσμάτων. Στην Εικόνα 1.3-1 απεικονίζονται σχηματικά οι τροχιές 24 δορυφόρων οι οποίοι βρίσκονται σε τροχιά γύρω από την γη. [5]



**Εικόνα 1.3-1: Τροχιές δορυφόρων γύρω από την Γη [5]**

Ωστόσο, το παραπάνω σύστημα, πάσχει από σοβαρούς περιορισμούς σε εσωτερικούς χώρους. Το γεγονός αυτό οφείλεται στο φαινόμενο απώλειας σημαντικής ισχύος από την συσκευή GPS εξαιτίας της εξασθένισης του σήματος. Αυτή προκαλείται λόγω της πολυπλοκότητας του εσωτερικού χώρου καθώς αυτός περιβάλλεται από πολλά αντικείμενα και έτσι υπάρχουν παρεμβολές και ανακλάσεις σήματος. Την ίδια στιγμή, πρόβλημα αποτελεί η αδυναμία του σήματος να διαπεράσει στερεά εμπόδια όπως τα κτήρια. [3]

Για την αντιμετώπιση αυτών των ζητημάτων έχουν παρουσιαστεί διάφορες τεχνολογίες. Η διαμόρφωση συχνότητας, η ραδιοφωνική αναγνώριση συχνότητας, το Wi-fi, το Bluetooth και αισθητήρες υπερ-ευρυζωνικότητας(UWB) αποτελούν παραδείγματα των τεχνολογιών αυτών. Παράλληλα, αξίζει να σημειωθεί ότι για ενίσχυση του εντοπισμού χρησιμοποιούνται υβριδικές προσεγγίσεις. [3]

## 1.4 Οργάνωση της διπλωματικής εργασίας

Η παρούσα διπλωματική εργασία επικεντρώνεται σε ένα drone εσωτερικού χώρου το οποίο κατασκευάστηκε εξολοκλήρου στο εργαστήριο του τομέα Κατεργασίας των Τεχνολογιών του Εθνικού Μετσόβιου Πολυτεχνείου και έχει ως σκοπό την πλοήγηση με την χρήση αισθητήρα υπερ-ευρυζωνικότητας(UWB).

Πιο συγκεκριμένα, σκοπός της διπλωματικής αποτελεί τόσο ο σχεδιασμός των ποδιών του εναέριου αεροσκάφους όσο και η εισαγωγή ενός προσομοιωμένου μοντέλου αυτού στο περιβάλλον του ROS(Robot Operating System). Με άλλα λόγια η μελέτη ξεκινά με την χρήση του σχεδιαστικού προγράμματος SolidWorks έτσι ώστε να αποτυπωθεί σε τρισδιάστατη μορφή το νέο σχέδιο των ποδιών του drone. Συνεχίζοντας, το μοντέλο iris διαθέσιμο από την εταιρεία PX4,εισάγεται στο περιβάλλον Gazebo όπου σε συνδυασμό με το σύστημα ROS μοντελοποιούνται όλοι οι αισθητήρες του drone και αυτό ακολουθεί μια καθορισμένη τροχιά.

Η δομή τους παρούσας εργασίας κατανέμεται σε εννέα κεφάλαια. Στο πρώτο κεφάλαιο πραγματοποιείται μια συνοπτική και γενική εισαγωγή. Αυτή περιλαμβάνει τον γενικό ορισμό των μη επανδρωμένων αεροσκαφών, την ιστορική τους αναδρομή και λεπτομέρειες για τον διαχωρισμό τους σε εσωτερικού και εξωτερικού χώρου. Το δεύτερο κεφάλαιο επικεντρώνεται στο drone που έχει κατασκευαστεί στο εργαστήριο. Αναλυτικότερα, πρόκειται να αναφερθούν τα δομικά του χαρακτηριστικά, η αλλαγή του από προηγούμενες μορφές, καθώς και ο νέος σχεδιασμός των ποδιών που προσαρμόζονται σε αυτό και στη βάση. Στη συνέχεια, το τρίτο κεφάλαιο, θα συμπεριλαμβάνει την δυναμική του εναέριου οχήματος, εισαγωγικές πληροφορίες για το πρόγραμμα Gazebo και το σύστημα ROS καθώς θα πραγματοποιηθεί και εκτενής αναφορά στον ελεγκτή της εταιρίας PX4. Έπειτα, το τέταρτο κατά σειρά κεφάλαιο της εργασίας θα περιλαμβάνει λεπτομερή αναφορά στον βασικό αισθητήρα του drone ο οποίος είναι ο αισθητήρας υπερ-ευρυζωνικότητας (UWB) αλλά και στο προσαρμοσμένο γυροσκόπιο. Επιπλέον, το πέμπτο κεφάλαιο παραθέτει μια σύντομη εξήγηση της λειτουργίας του κώδικα και όλων των αρχείων που τον πλαισιώνουν. Ακολούθως, το έκτο κεφάλαιο θα αποτελείται από όλες τις λεπτομέρειες της προσομοίωσης που υλοποιήθηκε στα πλαίσια της παρούσας εργασίας. Ολοκληρώνοντας, τα τελευταία τρία σε σειρά κεφάλαια θα περιλαμβάνουν τα συμπεράσματα, τις προτάσεις για μελλοντικές εργασίες και την βιβλιογραφία καθώς και ένα παράρτημα με τα απαραίτητα κατασκευαστικά σχέδια και τους κώδικες.

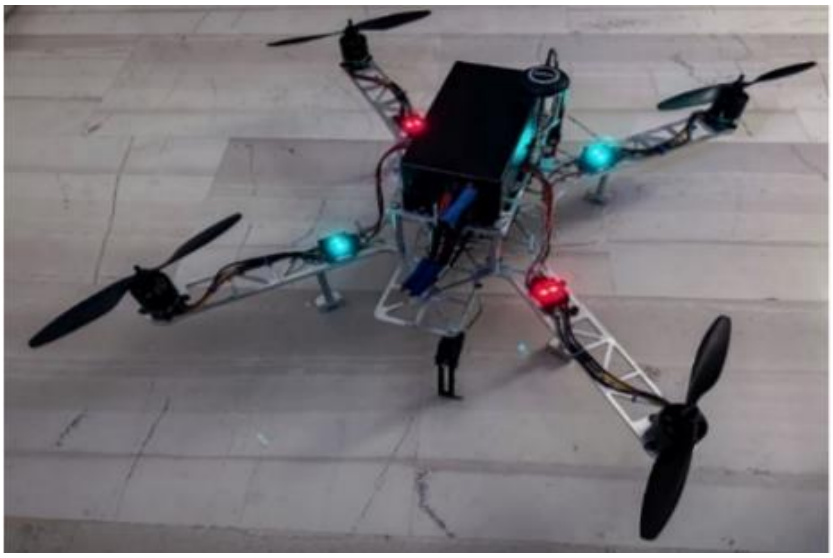
# 2 ΠΕΡΙΓΡΑΦΗ ΤΟΥ DRONE ΤΟΥ ΕΡΓΑΣΤΗΡΙΟΥ

Ξεκινώντας το παρόν κεφάλαιο αξίζει να σημειωθεί το γεγονός ότι το drone το οποίο θα συζητηθεί εκτενώς στην παρούσα διπλωματική εργασία έχει σχεδιαστεί και κατασκευασθεί εξολοκλήρου στο εργαστήριο του τομέα Τεχνολογίας των Κατεργασιών του Εθνικού Μετσόβιου Πολυτεχνείου.

## 2.1 Χαρακτηριστικά μη επανδρωμένου εναέριου οχήματος

### 2.1.1 Αρχικό drone εργαστηρίου

Στο εργαστήριο της σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείο κατασκευάσθηκαν δύο σκελετοί για το μη επανδρωμένο εναέριο όχημα. Το μοντέλο που αναπτύχθηκε αρχικώς απεικονίζεται στην Εικόνα 2.1-1.



**Εικόνα 2.1-1: Αρχικό drone εργαστηρίου [6]**

Πιο συγκεκριμένα, το πλαίσιο αυτού είναι κατασκευασμένο από αλουμίνιο της σειράς AI7075-T6. Η επιλογή του υλικού προκύπτει λόγω των εξαιρετικών του μηχανικών ιδιοτήτων οι οποίες δύναται να περιγραφούν μαθηματικά με τα μεγέθη της αντοχής σε εφελκυσμό ίση με 572 MPa και το όριο διαρροής το οποίο ισούται με 502 MPa. Παράλληλα, αξιοσημείωτα χαρακτηριστικά του εν λόγω υλικού αποτελούν η αντοχή σε κόπωση η οποία είναι ίση με 159 MPa, η καλή αντιδιαβρωτική συμπεριφορά και η αποδεκτή αντοχή σε κρούση που ισούται με 26 J. [7]

Όσον αφορά τις σχεδιαστικές παραμέτρους του αρχικού μη επανδρωμένου αεροσκάφους αποφασίστηκε η δημιουργία ενός συμμετρικού πλαισίου πάνω στο οποίο θα τοποθετηθούν όλα τα επιμέρους εξαρτήματα. Σκοπός αυτής της λογικής αποτελεί η μείωση του βάρους του drone έτσι ώστε να επιτευχθούν ικανοποιητικότεροι παράμετροι πτήσεις όπως ο αυξημένος χρόνος αυτής. Οι διαστάσεις του αεροσκάφους επιλέχθηκαν να είναι 700 x 700 x175 mm. Την ίδια στιγμή το βάρους του, υπολογίζοντας τόσο το πλαίσιο όσο και τα εξαρτήματα που απαρτίζουν το ρομποτικό σύνολο είναι ίσο με 5,9 kg και έχει την δυνατότητα μεταφοράς φορτίων 2 kg. Η μεταφορά του φορτίου θα γίνεται σε ειδικά καλάθια τα οποία έχουν κατασκευασθεί με την τεχνική της τρισδιάστατης εκτύπωσης και είναι δομημένα με υλικό ABS. Αυτά έχουν ποικιλία διαστάσεων όπως 200 x 80x50 και 70 x30 x30. [7]

Επιπλέον, αξιοσημείωτο ζητούμενο κατασκευής είναι και η κατανομή του χώρου των εξαρτημάτων με στόχο την ταύτιση του κέντρου μάζας με το κέντρο συμμετρίας του πλαισίου. Αυτό μειώνει τις πιθανότητες ανατροπής του drone ειδικότερα κατά την διάρκεια πτήσεων. Τα εξαρτήματα που έχουν

επιλεγεί για να στελεχώσουν το σύστημα του drone κοστίζουν συνολικά περίπου στα 1000 ευρώ και είναι επτά.

Επομένως, ξεκινώντας την αναφορά από το σύστημα κίνησης έχουν αγοραστεί τέσσερις κινητήρες από την εταιρία Cobra με κωδικό μοντέλου C-3520/12 (820), τέσσερις έλικες από την APC διαστάσεων 10x5 (in) οι οποίες συνδέονται άμεσα με τους κινητήρες και 4 ελεγκτές κινητήρων (ESC) από την εταιρία Hobbywing Skywalker με αριθμό μοντέλου 2-6S 60A UBEC. [8] Τα προαναφερόμενα εξαρτήματα και ο τρόπος τοποθέτησης αυτών μπορούν να φανούν αναλυτικά στην Εικόνα 2.1-2και στην Εικόνα 2.1-3.

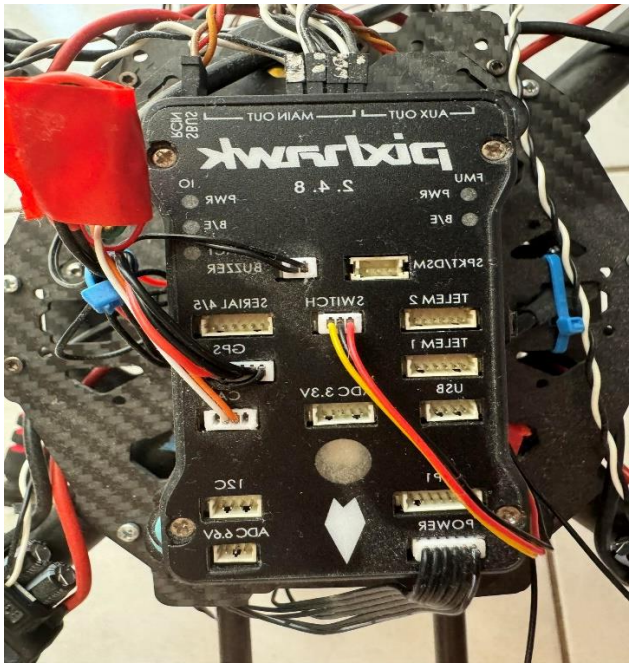

(α) (β)

**Εικόνα 2.1-2: (α) Ο κινητήρας του drone [8] και (β) Το ESC**

(α) (β)

**Εικόνα 2.1-3: (α) Οι έλικες και (β) Το συναρμολογημένο σύστημα κίνησης**

Ταυτόχρονα, καθοριστικά τμήματα του drone είναι και αυτά που είναι υπεύθυνα για την επιτυχία της πτήσης, της προσγείωσης, της απογείωσης και της διοχέτευσης ενέργειας. Αναλυτικότερα, ως ελεγκτής της πτήσης αγοράστηκε το μοντέλο PX4 2.4.8 με χωρητικότητα 32 bits από την εταιρία Pixhawk. Αυτός απεικονίζεται στην Εικόνα 2.1-4 και η λειτουργία του θα περιγραφεί αναλυτικά στη συνέχεια της παρούσας εργασίας.



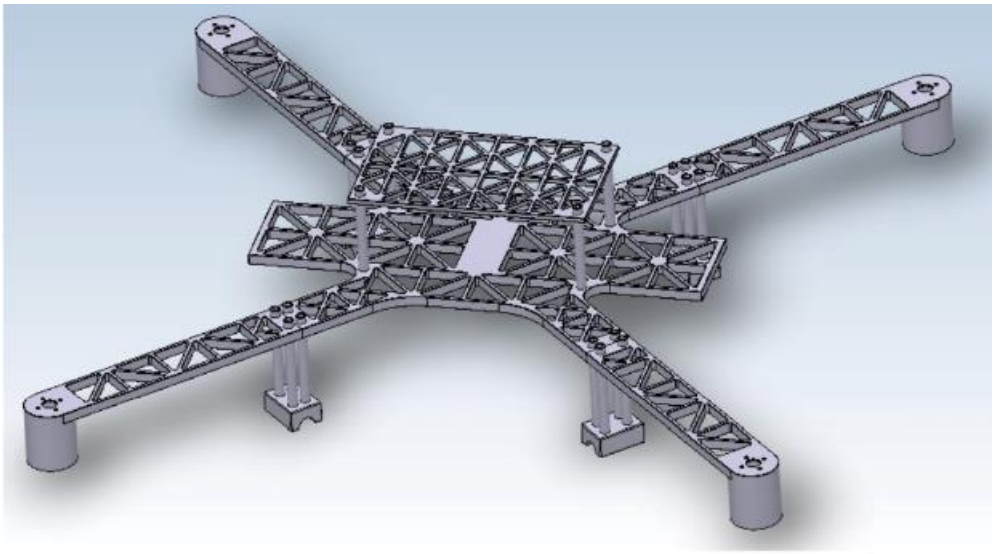

**Εικόνα 2.1-4: Ο ελεγκτής Pixhawk 2.4.8**

Επίσης, η μπαταρία που θα χρησιμοποιηθεί είναι από την URUAV με μοντέλο το XT90 και χρησιμοποιεί ως υλικό τα πολυμερή Λιθίου (Lipo) με χαρακτηριστικά τάσης 22,2V και λειτουργίας τις 10 Ah. Παράλληλα, απαραίτητο εξάρτημα είναι και ο φορτιστής ο οποίος λαμβάνεται από την εταιρία IMAX με μοντέλο B6 και δυνατότητας παροχής ισχύος 80 W και ρεύματος 6 A. Τέλος, στα εξαρτήματα προστίθεται και μια κάμερα από την εταιρία Eachine με μοντέλο E019 RC Drone 480P WiFi FPV η οποία είναι χρήσιμη για την διαδικασία της προσγείωσης [7]. Στην Εικόνα 2.1-5 μπορεί να παρατηρηθεί το εξάρτημα της μπαταρίας.


**Εικόνα 2.1-5: Η χρησιμοποιούμενη μπαταρία**

Πραγματοποιώντας αναφορά στην κατασκευή του drone, αυτό δημιουργήθηκε σε φρέζα CNC η οποία έχει διαδρομή 500 mm. Αυτό σημαίνει ότι το πλαίσιο πρέπει να είναι χωρισμένο σε ενότητες έτσι ώστε η

κάθε μία από αυτές να φορτώνεται στην τράπεζα της μηχανής. Το γεγονός αυτό αποτελεί πλεονέκτημα της κατασκευής καθώς διευκολύνεται η αντικατάσταση τμημάτων σε περίπτωση βλάβης, σύγκρουσης και πτώσης. Το πλαίσιο του αρχικού drone απεικονίζεται στην Εικόνα 2.1-6.



**Εικόνα 2.1-6: Πλαίσιο παλιού drone [7]**

### 2.1.2 Καινούργιο drone του εργαστηρίου

Λαμβάνοντας υπόψη την κατανομή του μεγέθους των drone όπως αυτή παρέχεται από την εταιρία JUOUAV και φαίνεται στην Εικόνα 2.1-7, το drone που κατασκευάζεται ανήκει στην κατηγορία των μικρών εναέριων οχημάτων. Αυτό συμβαίνει καθώς οι επιθυμητές διαστάσεις και το μήκος της προπέλας κατανέμονται στο συγκεκριμένο τύπο σύμφωνα με τον πίνακα. Την ίδια στιγμή δεν απαιτούνται περισσότεροι του ενός ανθρώπου έτσι ώστε να το μεταφέρουν γεγονός που το απομακρύνει περισσότερο από την κατηγορία των μεσαίων εναέριων οχημάτων.

| Size | Length | Propeller diameter | Weight | Use |
|---|---|---|---|---|
| Very small drones | 150mm (15cm, 6 inches) or less | 51mm (2 inches) or less | 200 grams (0.2kg, 0.44lbs) or less | Military surveillance |
| Small drones | Up to 300mm (12 inches) | 76-152mm (3-6 inches) | 200-1000 grams (0.44-2.2lbs) | • Indoor equipment inspections<br>• Recreation and photography |
| Medium drones | 300-1200mm (12 inches – 4 feet) | 150-640 mm (6-25 inches) | 1-20kg (2.2-44 pounds) | • Professional applications<br>• Amateur photography |
| Large drones | 120cm (4 feet) and up | 64 cm (25 inches) and up | 20kg (44 pounds) and up | • Enemy detection and combat capabilities<br>• Civil applications such as drone deliveries or filmmaking |

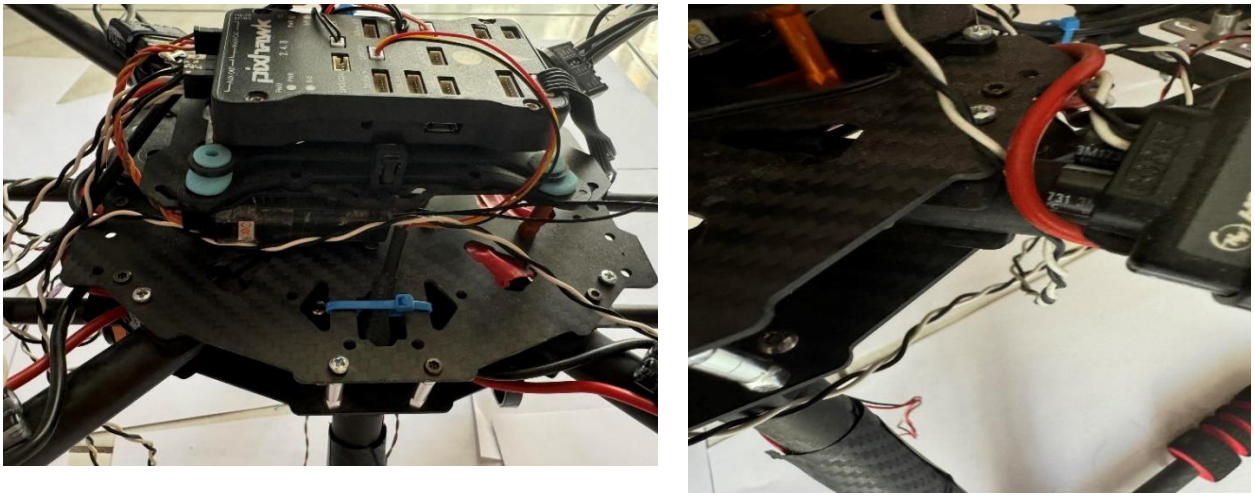
**Εικόνα 2.1-7: Πίνακας κατανομής μεγέθους των drones [9]**

Έχοντας υπόψη το παραπάνω το drone που κατασκευάσθηκε είναι πολύ βαρύ για την κατηγορία στην οποία υπάγεται. Το γεγονός αυτό οδηγεί σε ένα νέο σχεδιασμό. Βασική και μοναδική τροποποίηση αποτελεί το πλαίσιο το οποίο άλλαξε υλικό και σχήμα. Πιο συγκεκριμένα, το καινούργιο drone του εργαστηρίου απεικονίζεται στην Εικόνα 2.1-8.

**Εικόνα 2.1-8: Καινούργιο drone εργαστηρίου**

Το υλικό μεταβλήθηκε από αλουμίνιο της σειράς Al7075-T6 σε ανθρακόνημα (carbon fiber). Το υλικό αυτό προτιμάται στις αεροπορικές εφαρμογές καθώς δύναται να αντικαθιστά κράματα αλουμινίου και τιτανίου εξοικονομώντας βάρος[10]. Ωστόσο, έχει αναλογία αντοχής προς βάρος ίση με 3026 MPa/g/cm$^3$ η οποία είναι πολύ μεγαλύτερη από εκείνη ενός τυπικού αλουμινίου που ισούται με 115 MPa/g/cm$^3$.Με αυτόν τον τρόπο, οι ικανότητές του ενισχύονται. Παράλληλα, χαρακτηρίζεται από υψηλή αντοχή σε εφελκυσμό με το αντίστοιχο μέγεθος να ισούται με 5407 MPa [11]. Αυτό συμβαίνει καθώς οι ίνες του άνθρακα κατατάσσονται στις ισχυρότερες εμπορικές ίνες. Σημειώνοντας περισσότερες θετικές ιδιότητες, το ανθρακόνημα χαρακτηρίζεται από χαμηλή θερμική διαστολή και εξαιρετική αντοχή σε κόπωση συγκριτικά με τα μέταλλα [12].

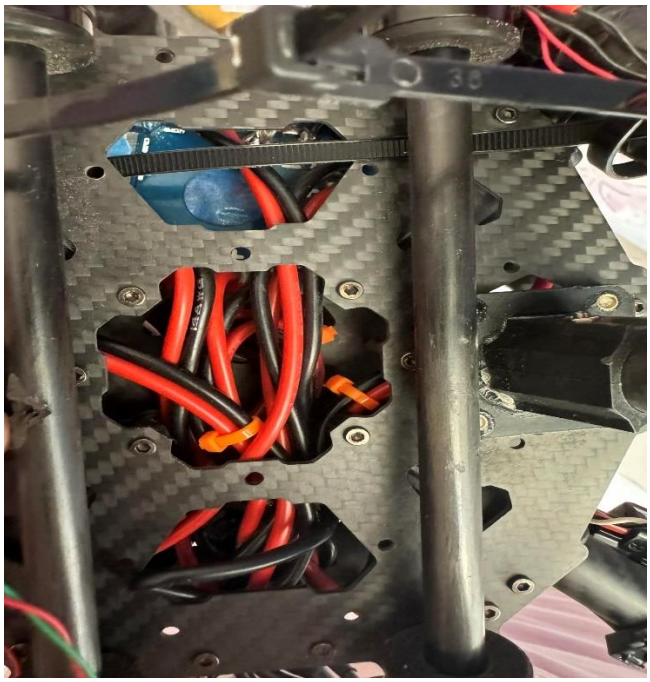
Μεταβολές παρατηρούνται και στο σχήμα του πλαισίου. Αναλυτικότερα, η λογική του τετρακοπτέρου παραμένει αλλά αφαιρούνται τμήματα με καμία χρησιμότητα. Η καινούργια λογική κατασκευής χαρακτηρίζεται από την δημιουργία δύο τμημάτων που λαμβάνουν την μορφή ορόφων. Άξιο αναφοράς είναι το κάτω τμήμα όπου ενσωματώνονται οι αρχές των στύλων οι οποίοι θα περιλαμβάνουν τελικά το σύστημα των κινητήρων. Τα δύο αυτά χαρακτηριστικά παρουσιάζονται στην Εικόνα 2.1-9.



(α) (β)

**Εικόνα 2.1-9: (α) Το νέο πλαίσιο (β) Η σύνδεση με τους ράβδους των κινητήρων**

Επιπλέον, το συγκεκριμένο τμήμα διαθέτει ειδική διαμόρφωση έτσι ώστε να μπορούν να προσαρμόζεται το φορτίο που θα μεταφερθεί από το εναέριο όχημα. Η διαμόρφωση αυτή παρατηρείται στην Εικόνα 2.1-10.



**Εικόνα 2.1-10: Το κάτω μέρος του drone με ειδική διαμόρφωση**

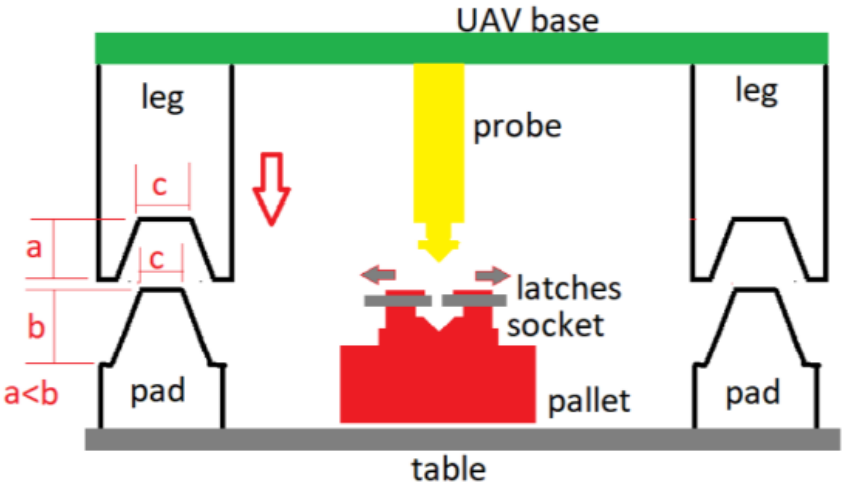
Η ανακατασκευή του πλαισίου και η διαμόρφωση δύο ορόφων έχει ένα ακόμα πλεονέκτημα. Αυτό είναι η μείωση του μεγέθους ολόκληρου του drone καθώς περιορίζονται οι διαστάσεις του. Πιο συγκεκριμένα, αυτές ανέρχονται σε 545x545x235 mm έναντι των τιμών 700x700x175 mm. Αξιοσημείωτη είναι η αύξηση του ύψους η οποία προκαλείται από την αλλαγή λογικής σχεδιασμού.

### 2.1.3 Σχεδιασμός και κατασκευή ποδιών του drone

Απαραίτητο μέρος για την επίτευξη της ακριβής τοποθέτησης του drone κατά την διαδικασία της προσγείωσης σε έναν σταθμό με στόχο τη φόρτωση ή την εκφόρτωση εξαρτημάτων αποτελούν τα πόδια του. Αυτά χωρίζονται σε δύο μέρη με το ένα να βρίσκεται ενσωματωμένο στο όχημα και το δεύτερο στερεωμένο στη βάση.

Ξεκινώντας την αναφορά από τα πόδια του αρχικού εναέριου οχήματος αυτά σχεδιάστηκαν σε τραπεζοειδές σχήμα. Πιο συγκεκριμένα, στόχος του εν λόγω σχεδιασμού αποτελεί η ομαλότερη διαδικασία προσγείωσης

με την μείωση των ταλαντώσεων αυτής. Βασική λεπτομέρεια αποτελεί η μειωμένη διάσταση εσοχής στο θηλυκό τμήμα σε σχέση με το ύψος του αρσενικού μέρους. Σκοπός αυτής της διαφοράς διαστάσεων είναι η ακριβέστερη τοποθέτηση του οχήματος καθώς αυτό ελαττώνει το ύψος του. Παράλληλα, κατά την διαδικασία του σχεδιασμού επιδιώκεται οι διαμήκεις οριζόντιοι άξονες των τραπεζοειδών τμημάτων των δύο αντί-διαμετρικών ποδιών του drone να είναι κάθετοι μεταξύ τους. Αυτό συμβαίνει γιατί, η απόσταση επιτυγχάνεται αμοιβαία σε δύο διευθύνσεις εξασφαλίζοντας ακόμα πιο ακριβή τοποθέτηση. Στην Εικόνα 2.1-11 παρουσιάζεται σχηματικά ο σχεδιασμός των ποδιών, η λειτουργία τους και οι διαστασιολογικές διαφορές. [7]

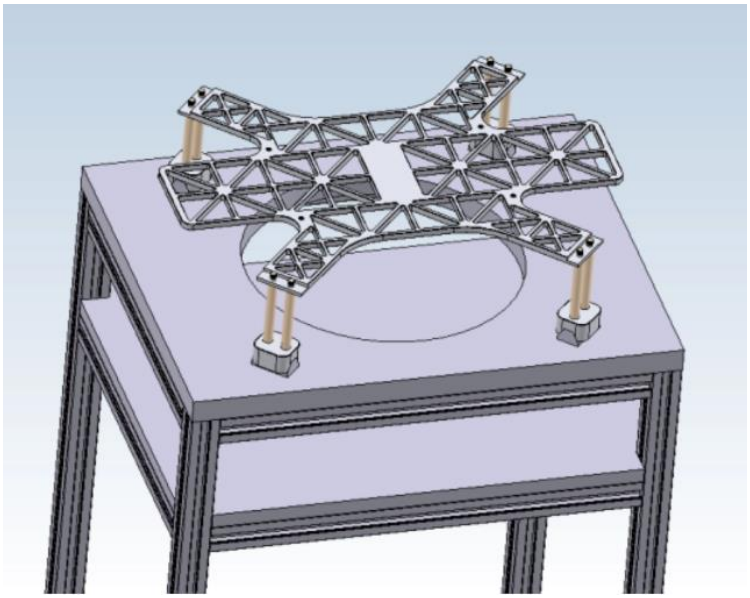



**Εικόνα 2.1-11: Σχηματική αναπαράσταση των ποδιών [7]**

Την ίδια στιγμή, στην Εικόνα 2.1-12 απεικονίζονται τα πόδια τα οποία κατασκευάσθηκαν στο εργαστήριο. Αυτά είναι από αλουμίνιο σειράς AA 7075-T6 από το οποίο έχει κατασκευασθεί το αρχικό drone. Επίσης, για την κατασκευή τους αξιοποιήθηκε μια φρέζα τύπου CNC.

**Εικόνα 2.1-12: Παλιά πόδια από αλουμίνιο**

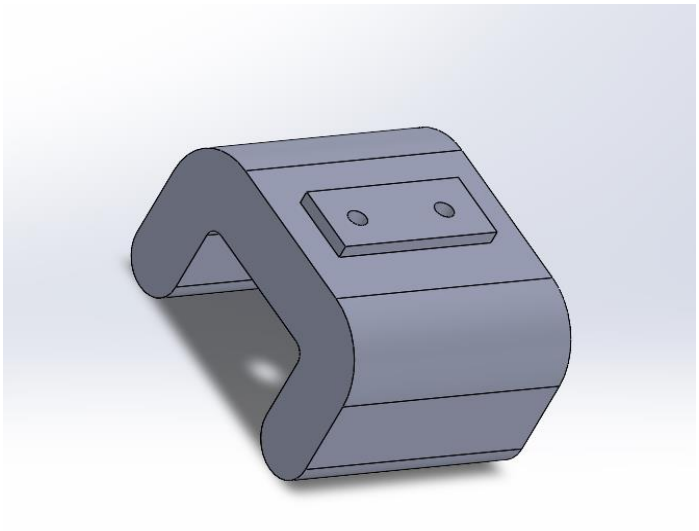
Επιπλέον, αξίζει να σημειωθεί ότι η συναρμογή μεταξύ των ποδιών και του βραχίονα υλοποιείται με βάση τους κοχλίες σύνδεσης. Πιο αναλυτικά, η άκρη του βραχίονα του drone διαθέτει δύο εσοχές για κοχλίες μεγέθους M6. Αυτοί συνδέονται στο κάτω μέρος τους με το πόδι του εναέριου οχήματος. Το γεγονός αυτό παρουσιάζεται στην Εικόνα 2.1-13. Επίσης, σημαντική παρατήρηση αποτελεί το φαινόμενο ότι εξαιτίας της διαδικασίας και του υλικού κατασκευής του αρχικού drone προτιμάται η εφαρμογή της κοχλιοσύνδεσης για όλες τις επιμέρους συνδέσεις.



**Εικόνα 2.1-13: Σύνδεση ποδιών με βάση [7]**

Η δημιουργία του καινούργιου εναέριου οχήματος επέφερε αλλαγές στην κατασκευή των ποδιών του. Αναλυτικότερα, αυτά επιλέχθηκαν να έχουν σφαιρικό σχήμα σε αντίθεση με το προηγούμενο τραπεζοειδές. Με στόχο το καλύτερο δυνατό αποτέλεσμα, δημιουργήθηκαν και υλοποιήθηκαν δύο σχεδιασμοί.

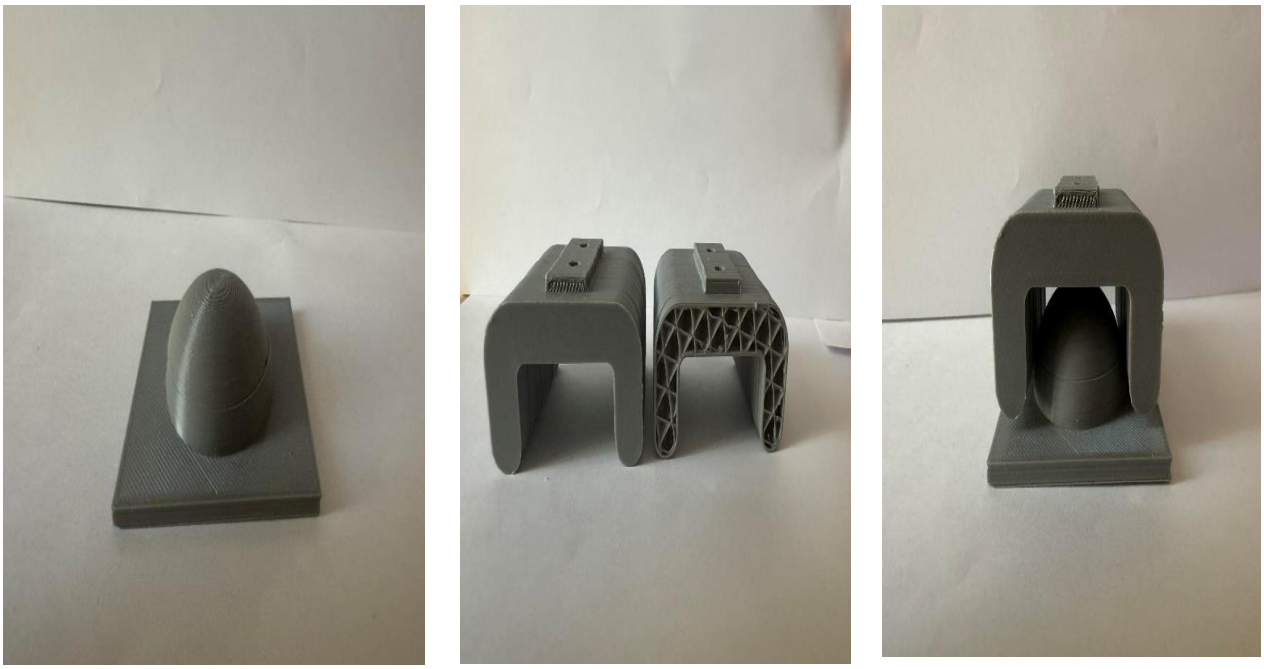
Ξεκινώντας με τον πρώτο, αποπειράθηκε η σχεδίαση ενός ποδιού κοίλης ορθογωνικής διατομής το οποίο θα διαθέτει στρογγυλοποιημένες άκρες. Στόχος της εν λόγω κατασκευής αποτελεί η εφαπτομενική εισχώρησή της στην καινούργια βάση. Αυτή θα έχει πλέον την μορφή σφαίρας. Για καλύτερη κατανόηση, στην Εικόνα 2.1-14 παρουσιάζεται το νέο πόδι το οποίο δημιουργήθηκε με την βοήθεια του προγράμματος SolidWorks.



**Εικόνα 2.1-14: Τρισδιάστατη όψη του αρχικού σχεδιασμού ποδιού**

Τον σχεδιασμό ακολούθησε και η κατασκευή των νέων τμημάτων. Πιο συγκεκριμένα, αυτά προτιμήθηκαν να υλοποιηθούν με την τεχνική της τρισδιάστατης εκτύπωσης. Αυτή η μέθοδος αποτελεί καλύτερη επιλογή στην παρούσα περίπτωση καθώς εξοικονομεί χρόνο και χρήμα. Παράλληλα, η ακρίβεια η οποία είναι απαραίτητη για την επιτυχία της συναρμογής εξασφαλίζεται με την προσαρμογή των παραμέτρων της εκτύπωσης όπως η ταχύτητα και το ύψος στήλης. Το υλικό που χρησιμοποιήθηκε είναι το PLA και το πρόγραμμα στο οποίο ανατέθηκε το τεμάχιο είναι το Ultimaker Cura το οποίο υποστηρίζεται από τον εκτυπωτή τρισδιάστατης εκτύπωσης μοντέλου Ended 3 neo.

Η διαδικασία της εκτύπωσης πραγματοποιήθηκε με ακροφύσιο διαμέτρου 0.8 mm και νήματος διαμέτρου 1.35 mm. Τα τεμάχια που δημιουργήθηκαν απεικονίζονται στην Εικόνα 2.1-15. Σε αυτή φαίνονται τόσο το κάθε ένα ξεχωριστά όσο και η μεταξύ τους σύνδεση.

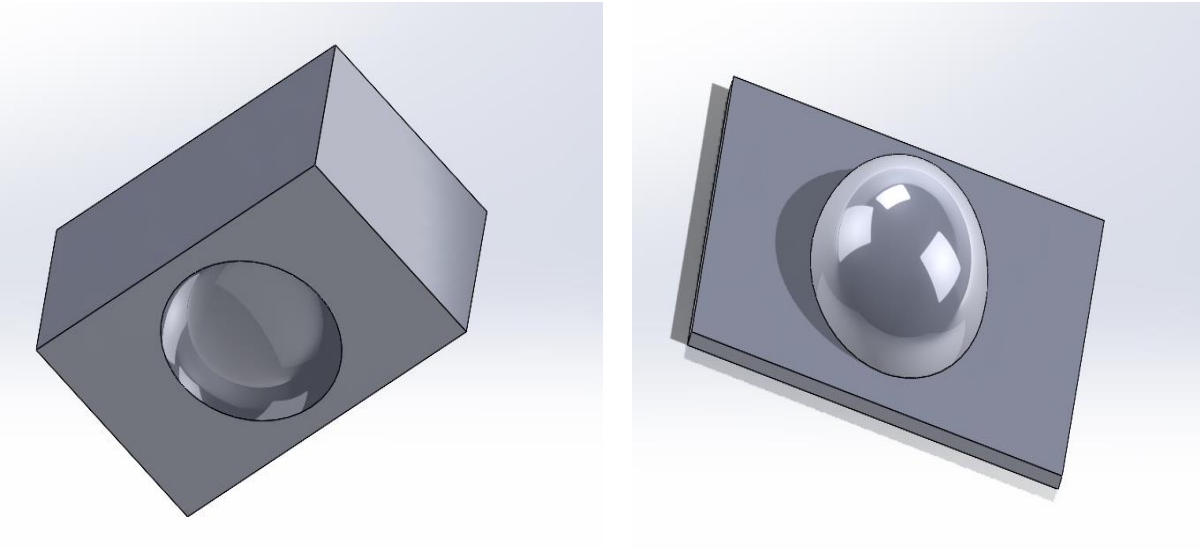


(α) (β) (γ)

**Εικόνα 2.1-15: Τα κατασκευασμένα τμήματα του ποδιού (α) Η βάση του (β) Το τμήμα του ποδιού και (γ) Η συναρμολόγησή τους**

Ωστόσο, εξαιτίας του γεγονότος ότι το drone θα μειώνει το κατακόρυφο ύψος του κατά την διαδικασία της προσγείωσης, η εφαπτομενική εισχώρηση εγκυμονεί προβλήματα επαφής καθώς το όχημα δύναται να μην τοποθετηθεί σωστά και να χάσει την ισορροπία του. Την ίδια στιγμή, οι διαστάσεις που προτιμήθηκαν στην αρχική σχεδίαση κρίνονται πολύ μεγαλύτερες από τις απαραίτητες. Ο συνδυασμός αυτών των προβλημάτων οδήγησε στην ανάγκη ενός νέου ποδιού.

Ο καινούργιος σχεδιασμός υιοθετεί και πάλι την λογική των τραπεζοειδών ποδιών με αλλαγή, όμως, του σχήματος. Πιο συγκεκριμένα, το πόδι το οποίο θα ενσωματώνεται στο drone έχει το σχήμα κοίλης σφαίρας ενώ η βάση διατηρείται ως έχει με μια απλή μείωση των διαστάσεων της. Ταυτόχρονα, το βάθος της κοίλης σφαίρας είναι μικρότερο από το ύψος της βάσης της. Το γεγονός αυτό επιτρέπει και πάλι την καλύτερη τοποθέτηση κατά την διάρκεια της προσγείωσης και αποτρέπει τις ανεπιθύμητες ταλαντώσεις. Τα νέα πόδια παράχθηκαν με την χρήση του σχεδιαστικού προγράμματος SolidWorks και παρουσιάζονται στην Εικόνα 2.1-16. Σε αυτή φαίνεται αφενός το εξάρτημα που θα προσαρμοστεί στο όχημα και αφετέρου η νέα βάση.

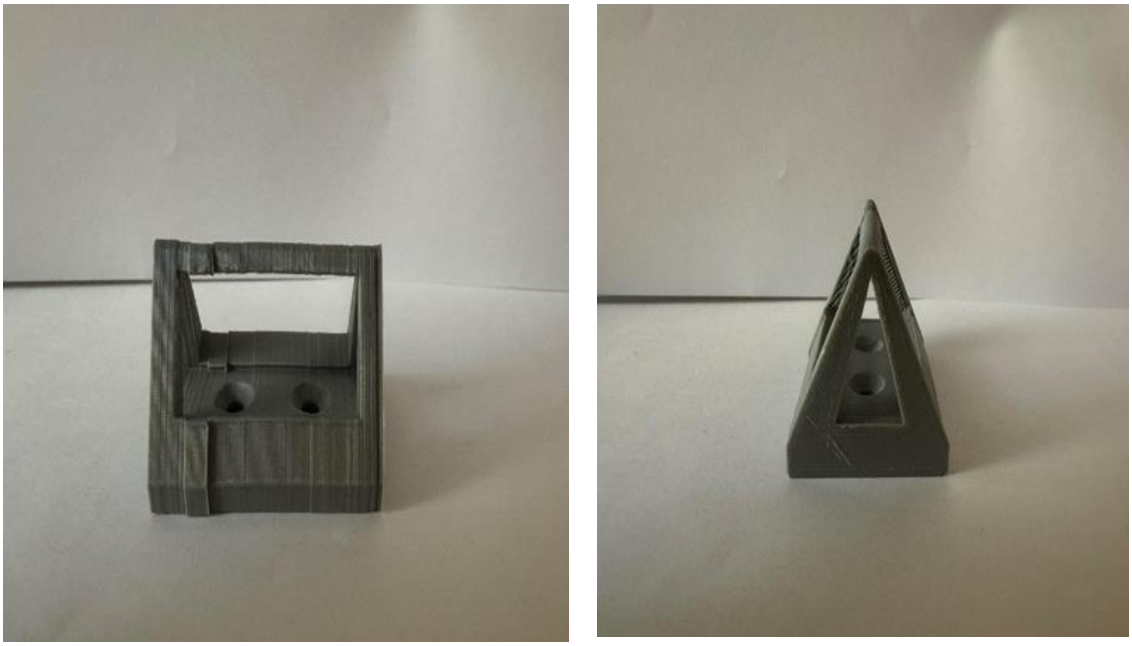


(α) (β)

**Εικόνα 2.1-16: Η καινούργια λογική των ποδιών (α) Το εξάρτημα που θα προσαρμοστεί στο drone και (β) Η νέα βάση**

Άξια παρατήρησης είναι η διαδικασία συναρμογής των νέων τμημάτων με το εναέριο όχημα. Αυτή την φορά, η επιλογή της κοχλιωτής σύνδεσης μεταξύ του βραχίονα του οχήματος και του ποδιού καθίστανται αδύνατη εξαιτίας του ανθρακονήματος το οποίο αποτελεί το νέο υλικό κατασκευής του drone. Αυτό, λόγω της παρουσίας των ινών άνθρακα και της συμπαγής μορφής του σχήματος της δοκού δεν παρέχει την δυνατότητα διάνοιξης οπών. Το φαινόμενο αυτό δύναται να λυθεί με την κατασκευή ειδικής διαμόρφωσης τριγωνικής διατομής η οποία θα προσαρμόζεται εφαπτομενικά στην δοκό και θα συγκρατείται στο απαραίτητο σημείο με την βοήθεια δομικής εποξικής κόλλας. Αυτή προμηθεύεται από την εταιρία 3M Scotch-Weld, έχει αριθμό τεμαχίου EC-9323 B/A Part A και βασικό συστατικό της είναι η τροποποιημένη αμίνη [13].

Η σύνδεση της διαμόρφωσης με τα υπόλοιπα εξαρτήματα του ποδιού γίνεται μέσω κοχλιών. Αναλυτικότερα, χρησιμοποιούνται 2 κοχλίες με φρεζάτη κεφαλή μεγέθους M4. Η επιλογή αυτή λαμβάνεται για πρακτικούς λόγους καθώς θα πρέπει να εισαχθούν στο εσωτερικό της τριγωνικής διατομής. Αυτή, υλοποιήθηκε, και πάλι, με την διαδικασία της τρισδιάστατης εκτύπωσης και χρήση υλικού PLA. Η κατασκευή έλαβε χώρα στο ίδιον ακριβώς εκτυπωτή με τις ίδιες παραμέτρους εκτύπωσης όπως και τα τεμάχια του πρωταρχικού σχεδιασμού. Στην Εικόνα 2.1-17 φαίνεται το εξάρτημα που κατασκευάσθηκε. Σε αυτό παρατηρείται μια σχεδιαστική ατέλεια η οποία οφείλεται στην σταθερότητα της διάταξης του εκτυπωτή.



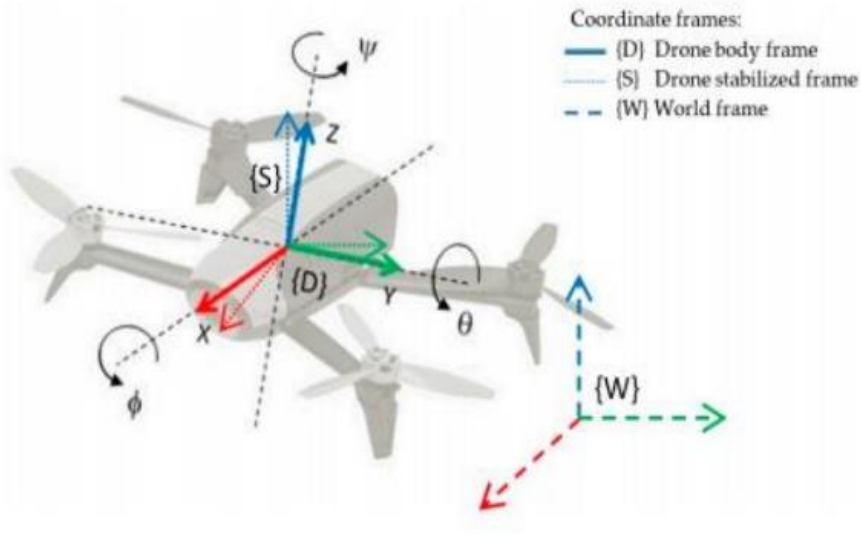
(α) (β)

**Εικόνα 2.1-17: Η κατασκευασμένη ειδική διαμόρφωση σε τριγωνική μορφή (α) Μπροστινή όψη (β) Πλάγια όψη**

# 3 ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΣΟΜΟΙΩΣΕΩΝ

## 3.1 Κινηματική του drone

Σημαντικό για την συνέχεια της εργασίας είναι η αποτύπωση εξισώσεων που διέπουν την κινηματική του εναέριου οχήματος. Ξεκινώντας, βασικό όρισμα αποτελούν τα συστήματα συντεταγμένων. Πιο αναλυτικά, αυτά είναι δύο ειδών, το χωρόδετο σύστημα αναφοράς (W) και το σωματόδετο σύστημα (S). Η διαφορά τους έγκειται στο γεγονός ότι το πρώτο βρίσκεται σταθερό στο περιβάλλον ενώ το δεύτερο ξεκινά από το κέντρο βάρους του drone και είναι παράλληλο με το δάπεδο. Αυτά απεικονίζονται σχηματικά στην Εικόνα 3.1-1Εικόνα 3.1-1: Συστήματα συντεταγμένων .




**Εικόνα 3.1-1: Συστήματα συντεταγμένων [8]**

Κατά την πτήση ενός εναέριου οχήματος στον χώρο ορίζονται έξι βαθμοί ελευθερίας. Αυτό σημαίνει ότι εκτός από τις μετακινήσεις στους άξονες x, y,z οι οποίες καθορίζουν την κατεύθυνση υπάρχουν και στροφές του οχήματος οι οποίες σχετίζονται με τον προσανατολισμό του. Αυτές ορίζονται με βάση της γωνίες Euler και χαρακτηρίζονται ως roll, pitch και yaw. Επομένως, σχηματίζονται δύο διανύσματα τα οποία δείχνουν την θέση του drone και είναι τα $x = (x, y, z)$ και $\theta = (\varphi, \theta, \psi)$ όπου με ϕ συμβολίζεται η κίνηση roll, με θ η κίνηση pitch και με ψ η κίνηση yaw. Την ίδια στιγμή, τα αντίστοιχα διανύσματα των ταχυτήτων προκύπτουν με απλή παραγώγιση και είναι τα $\dot{x} = (\dot{x}, \dot{y}, \dot{z})$ και $\dot{\theta} = (\dot{\varphi}, \dot{\theta}, \dot{\psi})$. [8]

Με στόχο την κατάστρωση των εξισώσεων οι οποίες θα περιγράφουν την κινηματική του drone εφαρμόζεται η διατύπωση των Euler-Lagrange. Έχοντας ως δεδομένο ότι στο σύστημα αναφοράς η επιτάχυνση προκύπτει εξαιτίας της ώσης, της βαρύτητας και της αεροδυναμικής τριβής λαμβάνεται η σχέση:

$$m\ddot{x} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \cdot T_B + F_D \qquad (3.1)$$

Στην σχέση (3.1) ως m συμβολίζεται η μάζα του οχήματος και ως g η επιτάχυνση της βαρύτητας. Επίσης, τα μεγέθη $F_D$ και $T_B$ αναπαριστούν την αντίσταση του αέρα και την δύναμη της ώσης στο σωματόδετο σύστημα αντίστοιχα. Επιπλέον, άξιο αναφοράς είναι το γεγονός ότι στην εξίσωση μπορούν να προστεθούν και άλλοι όροι που προσομοιώνουν αεροδυναμικά φαινόμενα, όπως η ελαστικότητα των ελίκων και οι επιδράσεις της ροής του αέρα από τους τοίχους και το έδαφος. Ωστόσο, η επίδραση αυτών των πρόσθετων όρων στα τελικά αποτελέσματα είναι ελάχιστη. Τέλος, το μέγεθος R είναι το μητρώο μετασχηματισμού το οποίο προκαλεί την μετάβαση από το ένα σύστημα συντεταγμένων στο άλλο [8]. Το μητρώο αυτό δίνεται από την σχέση (3.2) και είναι απαραίτητο στην παρούσα εξίσωση καθώς αυτή αναφέρεται στο χωρόδετο σύστημα συντεταγμένων.

$$R = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\varphi s_\varphi - c_\varphi s_\varphi & c_\psi s_\theta c_\varphi + s_\theta s_\psi \\ c_\theta s_\varphi & s_\varphi s_\theta s_\psi + c_\varphi c_\varphi & s_\varphi s_\theta c_\varphi - c_\varphi s_\theta \\ -s_\theta & c_\theta s_\varphi & c_\theta c_\varphi \end{bmatrix} \quad (3.2)$$

Παράλληλα, οι εξισώσεις οι οποίες περιγράφουν την περιστροφή δίνονται και πάλι στην μορφή Euler-Lagrange και είναι:

$$I\dot{\omega} + \omega \times (I\omega) = \tau \rightarrow$$
$$\omega = \begin{bmatrix} \tau_\varphi I_{xx}^{-1} \\ \tau_\theta I_{yy}^{-1} \\ \tau_\psi I_{zz}^{-1} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy} - I_{zz}}{I_{xx}} \omega_y \omega_z \\ \frac{I_{zz} - I_{xx}}{I_{yy}} \omega_x \omega_z \\ \frac{I_{xx} - I_{yy}}{I_{zz}} \omega_x \omega_y \end{bmatrix} \quad (3.3)$$

Στην σχέση (3.3) με $\tau_\phi$,$\tau_\theta$ και $\tau_\psi$ συμβολίζονται οι ροπές των roll, pitch και yaw αντίστοιχα ενώ με $I_{xx}$, $I_{yy}$ και $I_{zz}$ δίνονται οι ροπές αδράνειας στους αντίστοιχους άξονες.

Σε αυτό το σημείο αξίζει να αναφερθεί το γεγονός ότι η γωνιακή ταχύτητα ω δεν είναι ίση με $\dot{\theta}$. Αυτό συμβαίνει καθώς πρόκειται για δυο διαφορετικά μεγέθη. Με άλλα λόγια, η πρώτη απευθύνεται στο σωματόδετο σύστημα αναφοράς και είναι διάνυσμα με κατεύθυνση πάνω στον άξονα περιστροφής ενώ η δεύτερη αποτελεί απλή παράγωγος των γωνιών Euler [8].Ωστόσο, συνδέονται σύμφωνα με την σχέση:

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\varphi & c_\theta s_\varphi \\ 0 & -s_\varphi & c_\theta c_\varphi \end{bmatrix} \dot{\theta} \quad (3.4)$$

## 3.2 Σχετικά με το ROS

### 3.2.1 Εισαγωγή στο ROS

Το Robot Operating System (ROS) είναι ένα καινοτόμο πλαίσιο και σύνολο εργαλείων ανοικτού κώδικα που παρέχουν λειτουργικότητα ανάλογη με εκείνη ενός λειτουργικού συστήματος, ειδικά σχεδιασμένο για την υπηρεσιακή ρομποτική. Αναφέρεται ως εργαλείο το οποίο έχει κατορθώσει να προκαλέσει επανάσταση στον τομέα της ρομποτικής καθώς καθιστά την ανάπτυξη των εφαρμογών της πιο προσιτή, αποδοτική και ευέλικτη. Παρόλο που δεν είναι ένα παραδοσιακό λειτουργικό σύστημα, το ROS λειτουργεί ως μετά - λειτουργικό σύστημα, γεφυρώνοντας το κενό μεταξύ ενός λειτουργικού συστήματος και ενδιάμεσου λογισμικού. Προσφέρει μια πληθώρα χαρακτηριστικών που είναι κοινά σε λειτουργικά συστήματα, όπως αφαίρεση υλικού, διαχείριση ανταγωνισμού και διαχείριση διεργασιών, ενώ επιπλέον παρέχει ασύγχρονες και σύγχρονες κλήσεις, κεντρική βάση δεδομένων και σύστημα διαμόρφωσης ρομπότ. Με περισσότερα από 2000 πακέτα, κάθε ένα από τα οποία προσφέρει εξειδικευμένη λειτουργικότητα, το ROS αποτελεί μια ισχυρή πλατφόρμα για την ανάπτυξη εφαρμογών ρομποτικής. Παρότι η χρησιμότητά του δεν περιορίζεται αποκλειστικά στα ρομπότ, η κύρια δύναμή του έγκειται στη διαχείριση περιφερειακού υλικού, καθιστώντας το ένα απαραίτητο εργαλείο για ερευνητές και μηχανικούς ρομποτικής. [14]

### 3.2.2 Πλεονεκτήματα χρήσης του περιβάλλοντος ROS

Η ανάπτυξη του Robot Operating System (ROS) επέφερε μια σειρά από θετικά αποτελέσματα στον τομέα της ρομποτικής.

Ξεκινώντας, πριν από την εμφάνισή του, ο κάθε σχεδιαστής έπρεπε να αφιερώσει αξιοσημείωτο χρόνο και ενέργεια έτσι ώστε να δημιουργήσει λογισμικό για το ρομπότ το οποίο μελετούσε. Αυτή η διαδικασία απαιτούσε γνώσεις τόσο προγραμματισμού όσο και μηχανολογίας και ηλεκτρολογίας γεγονός που την έκανε απρόσιτη και δυσνόητη. Παράλληλα, τα προγράμματα που κατάφεραν να δημιουργηθούν ήταν στενά

συνδεδεμένα με το υποκείμενο υλικό, περιορίζοντας την επαναχρησιμοποίησή τους. Επομένως, το βασικό πλεονέκτημα του συστήματος είναι η αποφυγή της συνεχής επανεφεύρεσης του τροχού. Με άλλα λόγια, σκοπός είναι η προσφορά των βασικών λειτουργιών με τυποποιημένο τρόπο όπως ακριβώς συμβαίνει σε ένα συμβατικό λειτουργικό σύστημα για υπολογιστές. Το γεγονός αυτό θα επιτρέψει στους ερευνητές και τους μηχανικούς να επικεντρώνονται στις καινοτόμες πτυχές της ρομποτικής, αντί να επαναλαμβάνουν βασικές διαδικασίες και να δημιουργούν εξαρχής υποδομές λογισμικού. [15]

Ένα ακόμα κύριο πλεονέκτημα του ROS είναι η διευκόλυνση της ανάπτυξης ρομποτικών προσομοιώσεων το οποίο συρρικνώνει τον χρόνο και το κόστος της υλοποίησης πειραμάτων. Πιο αναλυτικά, οι ερευνητές και οι μηχανικοί χρησιμοποιούν πλέον το ROS και με αυτόν τον τρόπο δεν χρειάζεται να κατασκευάζουν συνέχεια καινούργιες διατάξεις για κάθε ένα έργο ρομποτικής. Αυτό έχει ως αντίκτυπο ένα σημαντικό οικονομικό όφελος καθώς ελαττώνεται κατά πολύ ο χρόνος διάθεσης των προϊόντων στην αγορά. Ταυτόχρονα, σημαντική πτώση καταγράφεται και στους πόρους που απαιτούνται για την ανάπτυξη νέων ρομποτικών εφαρμογών. [15]

Επιπλέον, το ROS επιτρέπει συνδυασμό τεχνογνωσίας από πληθώρα κλάδους. Πιο συγκεκριμένα, η δημιουργία ενός ρομποτικού συστήματος απαιτεί διαχείριση υλικού, μνήμης και διεργασιών, την αντιμετώπιση ζητημάτων παραλληλισμού και ταυτόχρονης χρήσης και την παροχή αλγορίθμων συλλογισμού. Εξαιτίας της δημιουργίας αυτού του λογισμικού όλα αυτά τα στοιχεία μπορούν να διαχειριστούν με ένα ενοποιημένο πλαίσιο, καθιστώντας πιο εύκολη τη συνεργασία μεταξύ διαφορετικών ειδικοτήτων, όπως η μηχανολογία, η πληροφορική και η τεχνητή νοημοσύνη. [15]

Τελευταίο αλλά εξίσου σημαντικό πλεονέκτημα του ROS είναι η μείωση των τεχνικών γνώσεων που απαιτούνται για την εργασία στον τομέα της ρομποτικής. Πιο αναλυτικά, με την χρήση του λογισμικού οι μηχανικοί αποκτούν την δυνατότητα να αξιοποιούν προ υπάρχοντα εργαλεία και βιβλιοθήκες. Με αυτόν τον τρόπο, δεν υπάρχει η ανάγκη εξειδικευμένων γνώσεων όλων των πτυχών της ρομποτικής επιστήμης. [15]

### 3.2.3 Χαρακτηριστικά του λογισμικού ROS

Όπως έχει αναφερθεί στα προηγούμενα υπό κεφάλαια η βασική αρχή ενός ρομποτικού λειτουργικού συστήματος, όπως είναι το ROS,είναι η παράλληλη εκτέλεση μεγάλου αριθμού εκτελέσιμων προγραμμάτων που πρέπει να μπορούν να ανταλλάσσουν δεδομένα συγχρονισμένα ή ασύγχρονα. Επιπλέον, η διαδικασία αυτή πρέπει να υλοποιείται συνεχώς και παράλληλα διασφαλίζοντας την αποτελεσματική πρόσβαση στους πόρους του εκάστοτε ρομπότ. Σε αυτό το κεφάλαιο θα πραγματοποιηθεί η απαραίτητη αναφορά σε έννοιες που χρησιμοποιούνται από το σύστημα καθώς το λογισμικό εκτελείται. Οι έννοιες αυτές είναι βασικές για την κατανόηση του κώδικα σε επόμενα κεφάλαια της διπλωματικής εργασίας.

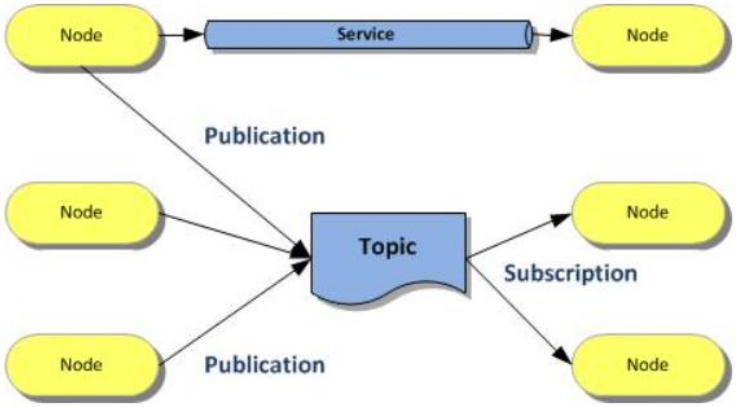
Ξεκινώντας, πραγματοποιείται αναφορά στην έννοια του κόμβου (node). Στο ROS ένας κόμβος ταυτίζεται με ένα εκτελέσιμο αρχείο. Με αυτόν τον τρόπο, αυτός ισοδυναμεί με έναν αισθητήρα, κινητήρα, αλγόριθμο επεξεργασίας κ.ο.κ. Κάθε κόμβος που αρχίζει να εκτελείται δηλώνει τον εαυτό του στον Κύριο(Master). Αυτό επιστρέφει στην αρχιτεκτονική του πυρήνα, σύμφωνα με την οποία κάθε πόρος είναι ένας ανεξάρτητος κόμβος. Εξαιτίας της λογικής των κόμβων οι βασικές λειτουργίες μπορούν να τυποποιηθούν και τα τεχνολογικά δομικά στοιχεία μπορούν να αναπτυχθούν γρήγορα και να επαναχρησιμοποιηθούν, να τροποποιηθούν ή να βελτιωθούν εύκολα. [15]

Συνεχίζοντας, σημαντικό είναι να αναφερθεί η έννοια του Κυρίου (Master). Αυτό είναι μια υπηρεσία δήλωσης και καταχώρησης κόμβων η οποία τους επιτρέπει να εντοπίζουν ο ένας τον άλλο και να ανταλλάζουν δεδομένα. Παράλληλα, η διάθεση αυτή των δεδομένων υλοποιείται μέσω ενός ευρέως χρησιμοποιούμενου συστατικού που ονομάζεται Parameter Server, το οποίο υλοποιείται με τη μορφή XMLRPC και είναι ένα είδος κεντρικής βάσης δεδομένων. Ολόκληρη η λειτουργία του Κυρίου υλοποιείται με τη χρήση XMLRPC. [15]

Ταυτόχρονα, άλλο ένα απαραίτητο σημείο είναι τα θέματα (Topics). Ένα θέμα είναι ένα σύστημα μεταφοράς δεδομένων που στηρίζεται σε ένα άλλο σύστημα δημοσίευσης. Ένας ή περισσότεροι κόμβοι είναι σε θέση να

δημοσιεύουν δεδομένα σε ένα θέμα και ένας ή περισσότεροι κόμβοι μπορούν να διαβάζουν δεδομένα από ένα θέμα. Επομένως, οι κόμβοι στέλνουν και λαμβάνουν μηνύματα σε θέματα. Με αυτόν τον τρόπο, ένα θέμα μπορεί να παρομοιαστεί με ένα ασύγχρονο δίαυλο μηνυμάτων. Αυτή η ιδιότητά του είναι απαραίτητη σε μια κατάσταση κατανεμημένου συστήματος. Το κάθε θέμα είναι τυποποιημένο γεγονός που σημαίνει ότι ο τύπος των δεδομένων που δημοσιεύονται είναι συνέχεια δομημένος με τον ίδιο τρόπο. [15]

Δύο, επιπλέον, συχνά αναφερόμενες έννοιες είναι αυτές των μηνυμάτων (Messages) και των υπηρεσιών (Services). Ως ένα μήνυμα ορίζεται μία σύνθετη δομή δεδομένων η οποία αποτελείται από έναν συνδυασμό πρωτογενών τύπων όπως ακέραιοι αριθμοί και συμβολοσειρές χαρακτήρων αλλά και από άλλα μηνύματα. Παράλληλα, μια υπηρεσία λύνει το πρόβλημα της σύγχρονης επικοινωνίας μεταξύ δύο κόμβων. Η ροή των δεδομένων μεταξύ όλων των εννοιών που αναφέρθηκαν παραπάνω απεικονίζεται στην Εικόνα 3.2-1. [15]




**Εικόνα 3.2-1: Γραφική απεικόνιση της ροής των δεδομένων μεταξύ των βασικών εννοίων του ROS [15]**

Στο σημείο αυτό αξίζει να σημειωθεί το γεγονός ότι οι προσομοιώσεις της παρούσας διπλωματικής εργασίας έγιναν με την χρήση της έκδοσης ROS Noetic Ninjemys . Αυτή είναι η δέκατη τρίτη έκδοση της εταιρίας και κυκλοφόρησε τον Μάρτιο του 2020. Για την χρήση της ήταν απαραίτητη η εγκατάσταση των Ubuntu της έκδοσης 20.04.

## 3.3 Το πρόγραμμα Gazebo

Σύμφωνα με το προηγούμενο κεφάλαιο το ROS είναι ένα σύνολο το οποίο περιλαμβάνει εργαλεία και αλγόριθμους τα οποία χρησιμοποιούνται κατά την διάρκεια προγραμματισμού, προσομοιώσεων και εκτέλεσης εργασιών στο ρομπότ. Ένα από αυτά τα εργαλεία το οποίο προσφέρει προσομοίωση στον τρισδιάστατο χώρο είναι αυτό του Gazebo.

Το Gazebo είναι μία εφαρμογή ανοιχτού κώδικα η οποία παρέχει ένα ρεαλιστικό περιβάλλον τρισδιάστατης προσομοίωσης. Αυτό επιτρέπει στους χρήστες να δημιουργούν εικονικούς κόσμους στους οποίους μπορούν να δοκιμάσουν τα ρομπότ τους πριν τα αναπτύξουν στον πραγματικό κόσμο. Επίσης, περιλαμβάνει τη δυνατότητα προσομοίωσης φυσικών αλληλεπιδράσεων, όπως η βαρύτητα, οι δυνάμεις επαφής και η τριβή, καθώς και η αλληλεπίδραση με διάφορα αισθητήρια συστήματα, όπως κάμερες και λέιζερ. Την ίδια στιγμή η εφαρμογή υποστηρίζει ένα ευρύ φάσμα ρομποτικών πλατφόρμων επιτρέποντας την απρόσκοπτη μετάβαση από την προσομοίωση στην πραγματική εφαρμογή. Με τη χρήση του Gazebo, οι προγραμματιστές μπορούν να βελτιστοποιήσουν τον σχεδιασμό των ρομπότ τους, να εκτελούν εκτεταμένες δοκιμές και να επιλύουν προβλήματα προτού τα ρομπότ φτάσουν σε πειραματικό στάδιο. Αυτό είναι κρίσιμο για τη μείωση του κόστους και του κινδύνου που σχετίζεται με την ανάπτυξη νέων ρομποτικών τεχνολογιών. [16]

Η ανάγκη για ύπαρξη προσομοίωσης στα ρομποτικά συστήματα υπό διάφορες συνθήκες και περιβάλλοντα οδήγησε στην ανάπτυξη της εφαρμογής. Η εκκίνηση της λογικής ξεκίνησε το φθινόπωρο του 2002 στο Πανεπιστήμιο της Νότιας Καλιφόρνιας από τον Dr. Andrew Howard και τον φοιτητή του, Nate Koenig. Ο

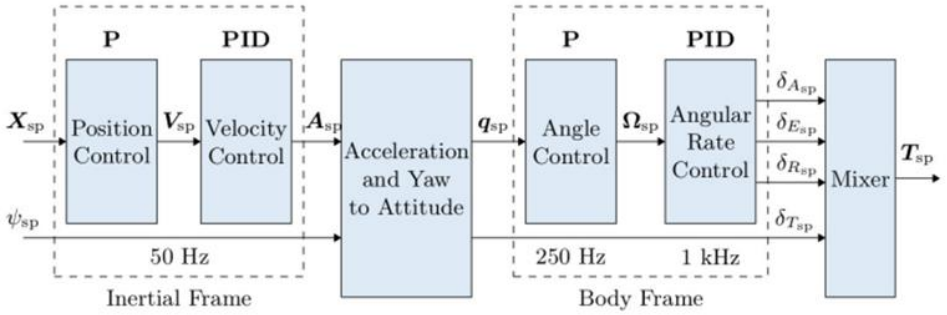
δεύτερος συνέχισε να αναπτύσσει το Gazebo κατά την διάρκεια του διδακτορικού του και το 2009 ο John Hsu, ανώτερος ερευνητής μηχανικός της Willow Garage, ενσωμάτωσε το ROS και το PR2 σε αυτό. Κατά την διάρκεια της πορείας του και έχοντας γίνει βασικό εργαλείο της κοινότητας του ROS τέθηκε υπό την διαχείριση της Open Source Robotics Foundation . Αυτή αναβάθμισε ακόμα περισσότερο την εφαρμογή και συνεχίζει την ανάπτυξή της μέχρι και σήμερα. [16]

## 3.4 Ο ελεγκτής της PX4

### 3.4.1 Η δομή του ελεγκτή του συστήματος

Κατά την εκτέλεση της παρούσας διπλωματικής εργασίας χρησιμοποιήθηκε ο ελεγκτής από την εταιρία PX4 με μοντέλο Pixhawk 2.4.8. Αυτό σημαίνει ότι δεν χρειάστηκε να δημιουργηθεί από την αρχή μια αρχιτεκτονική ελέγχου για την απογείωση και την ομαλή λειτουργία του drone σε σταθερή κατάσταση.

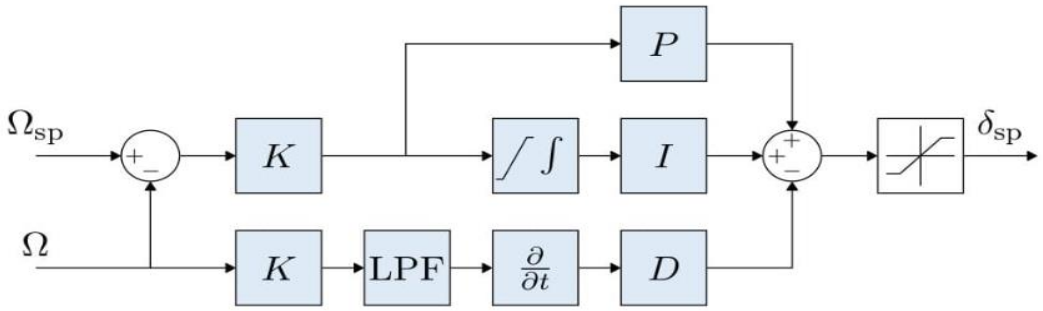
Πιο συγκεκριμένα, το συνολικό μοντέλο του ελεγκτή συνδυάζει PID και απλά P κυκλώματα όπου η ανατροφοδότησή τους σε κάθε ένα από τα εσωτερικά συστήματα υπολογίζεται μέσω ενός διευρυμένου φίλτρου Κάλμαν . Αυτό πραγματοποιεί τις εκτιμήσεις της θέσης και της ταχύτητας. Με αυτόν τον τρόπο, ο έλεγχος του drone ολοκληρώνεται με επιτυχία. Στην Εικόνα 3.4-1 παρουσιάζεται η συνολική αρχιτεκτονική του ελεγκτή με την μορφή δομικού διαγράμματος. Σημαντική παρατήρηση είναι το γεγονός ότι ανάλογα με τη λειτουργία, ο εξωτερικός βρόχος ο οποίος αναφέρεται στην θέση παρακάμπτεται και εμφανίζεται ύστερα από τον εξωτερικό βρόχο. Η χρήση του βρόχου θέσης είναι απαραίτητη μόνο όταν διατηρείται η θέση ή όταν η ζητούμενη ταχύτητα σε έναν άξονα είναι μηδενική. [17]




**Εικόνα 3.4-1: Συνολική αρχιτεκτονική του ελεγκτή σε μορφή δομικού διαγράμματος [17]**

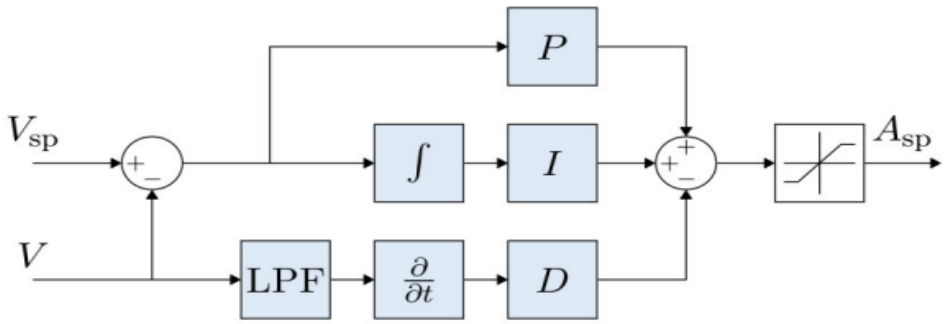
Παράλληλα, σημαντική κρίνεται και η διευκρίνιση των βασικών τμημάτων του δομικού διαγράμματος της Εικόνα 3.4-1.

Ξεκινώντας, αναφέρεται το υπό σύστημα του ελεγκτή του γωνιακού ρυθμού. Αυτό απεικονίζεται στην Εικόνα 3.4-2. Πιο αναλυτικά, αυτό αποτελείται από έναν K-PID ελεγχτή. Αυτός είναι μια μικτή υλοποίηση των δύο μαθηματικών ισοδύναμων μορφών του κλασσικού PID -της παράλληλης και της τυπικής- και παρέχει την δυνατότητα επιλογής μεταξύ τους, μέσω του ορίσματος του αναλογικού κέρδους K. Την ίδια στιγμή, ο ελεγκτής αποτελείται και από ένα χαμηλοπερατό φίλτρο με στόχο την μείωση του θορύβου. [17]



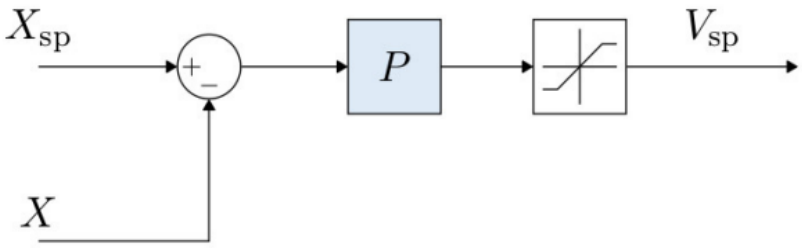

**Εικόνα 3.4-2: Δομικό διάγραμμα ελεγχτή γωνιακού ρυθμού [17]**

Συνεχίζοντας, γίνεται αναφορά στον ελεγχτή της ταχύτητας. Το δομικό διάγραμμά του παρατηρείται στην Εικόνα 3.4-3 και είναι όμοιο με ένα διάγραμμα ενός απλού ελεγχτή PID με χαμηλοπερατό φίλτρο. Ωστόσο, αξίζει να σημειωθεί ότι ο όρος της ολοκλήρωσης σε αυτό το διάγραμμα εφαρμόζεται κατευθείαν στην μέτρηση και όχι στο σφάλμα αυτής με στόχο την αποφυγή του σφάλματος ολοκλήρωσης, το οποίο είναι συχνό φαινόμενο σε αυτές τις διατάξεις. [17]




**Εικόνα 3.4-3: Δομικό διάγραμμα ελεγκτή ταχύτητας [17]**

Τέλος, παρατίθεται το διάγραμμα του ελέγχου της θέσης. Αυτό είναι ένας απλός ελεγκτής μορφής P και φαίνεται στην Εικόνα 3.4-4.

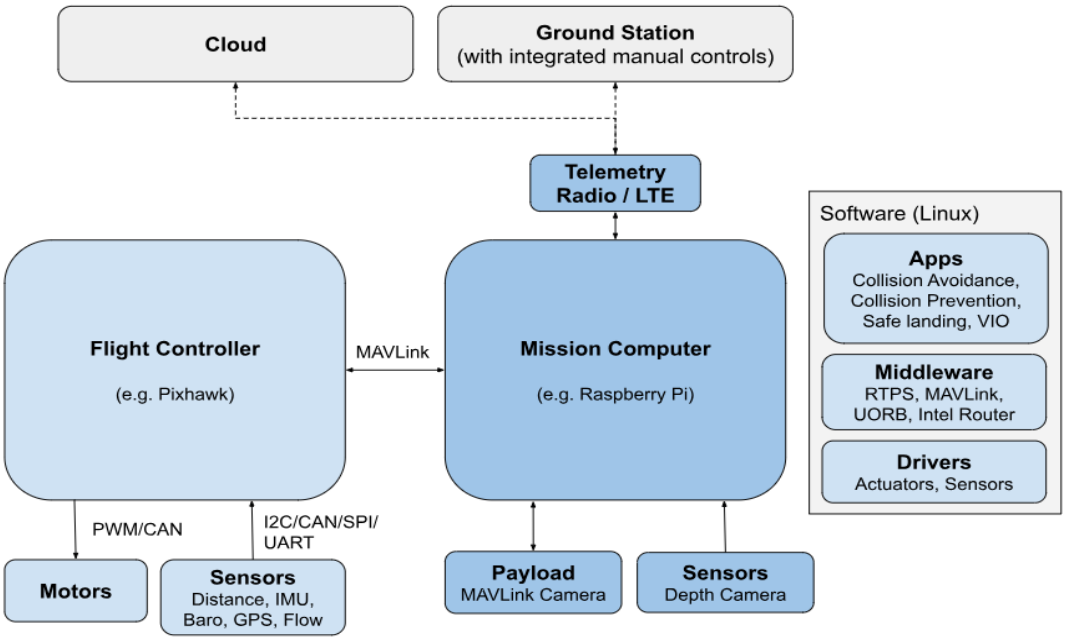



**Εικόνα 3.4-4: Δομικό διάγραμμα ελεγκτή θέσης [17]**

### 3.4.2 Αρχιτεκτονική του συστήματος

Το σύστημα το οποίο πρόκειται να δημιουργηθεί περιλαμβάνει αφενός τον ελεγκτή πτήσης της PX4 και αφετέρου έναν βοηθό υπολογιστή.

Η λειτουργία του πρώτου έγκειται στην εκτέλεση της πτήσης του συστήματος ενώ το δεύτερο είναι υπεύθυνο για περισσότερο εξειδικευμένες εντολές όπως η αποφυγή αντικειμένων και η πρόληψη συγκρούσεων. Τα δύο συστήματα επικοινωνούν χρησιμοποιώντας το πρωτόκολλο MAVLink και η συνολική αρχιτεκτονική τους φαίνεται στην Εικόνα 3.4-5. Σύμφωνα με αυτή γίνεται χρήση τηλεμετρίας έτσι ώστε να υπάρχει επικοινωνία με τον επίγειο σταθμό. Ταυτόχρονα, αξιοποιούνται πληθώρα αισθητήρων και μία πλακέτα ελεγκτή η οποία περιλαμβάνει πυξίδα, γυροσκόπιο και βαρόμετρο. Τέλος, σημαντικά στοιχεία είναι οι κεραίες ραδιοελέγχου οι οποίες επιτρέπουν τον χειροκίνητο χειρισμό του εναέριο οχήματος αλλά και η μονάδα ισχύος με σκοπό της την λειτουργία των κινητήρων. [18]




**Εικόνα 3.4-5: Αρχιτεκτονική του συστήματος του drone [18]**

Επιπλέον, αξίζει να σημειωθεί ότι το λειτουργικό σύστημα το οποίο χρησιμοποιείται για την εκτέλεση των λειτουργιών είναι το Linux καθώς χαρακτηρίζεται ως το καταλληλότερο για παρόμοιες εφαρμογές. Ταυτόχρονα, οι βασικοί αισθητήρες οι οποίοι θα σχολιαστούν στα πλαίσια της παρούσας διπλωματικής εργασίας είναι το γυροσκόπιο (IMU) και το Ultra Wideband (UWB) με το τελευταίο να αποτελεί το βασικό συστατικό της πλοήγησης του drone.

### 3.4.3 Οι διαφορές μεταξύ PX4-Autopilot και ArduPilot

Με στόχο την λειτουργία του ελεγχτή είναι απαραίτητη η εγκατάσταση και η φόρτωση σε αυτόν ενός ανοιχτού κώδικα λογισμικού το οποίο θα προσφέρει τα αναγκαία χαρακτηριστικά και δυνατότητες. Τα πιο γνωστά και ευρέως χρησιμοποιούμενα λογισμικά είναι δύο, το PX4-Autopilot και το ArduPilot.

Ξεκινώντας με το πρώτο, δύναται να προσφέρει σημαντική ακρίβεια και αξιοπιστία. Πιο αναλυτικά, διαθέτει αρθρωτή αρχιτεκτονική, η οποία επιτρέπει την προσαρμογή και την επέκταση της λειτουργικότητάς του. Με αυτήν την ευέλικτη αρχιτεκτονική, παρέχεται η δυνατότητα δημιουργίας και ενσωμάτωσης αυτοσχέδιων μονάδων, αισθητήρων και αλγορίθμων ελέγχου, γεγονός που το καθιστά λογισμικό εξαιρετικά προσαρμόσιμο για διάφορες πλατφόρμες και εφαρμογές drone [19].

Επιπλέον, το PX4-Autopilot διαθέτει προηγμένες δυνατότητες αυτόματου πιλότου, όπως υποστήριξη πολλών τρόπων πτήσης, αποφυγή εμποδίων και πλοήγηση χωρίς GPS. Αυτά τα χαρακτηριστικά το καθιστούν κατάλληλο για επαγγελματικές εφαρμογές. Η ικανότητά του να διαχειρίζεται αυτές τις απαιτητικές λειτουργίες με ακρίβεια και αξιοπιστία το καθιστά ιδανικό για σύνθετες και κρίσιμες αποστολές. [19]

Τέλος, η υιοθέτηση του λογισμικού PX4 από την βιομηχανία αποτελεί έναν σημαντικό λόγο επιλογής του. Αναλυτικότερα, αυτό έχει αναγνωριστεί από αρκετούς κατασκευαστές εναέριων οχημάτων εξαιτίας της σταθερότητας και την σιγουριά που δύναται να προσφέρει. [19]

Σε αντίθεση με το λογισμικό PX4-Autopilot, το ArduPilot δίνει μεγαλύτερη έμφαση στην ευελιξία και στην δημιουργία κοινότητας αλληλοϋποστήριξης. Πιο συγκεκριμένα, αυτό διέπεται από ένα ευρύ φάσμα υποστηριζόμενων πλατφόρμων οι οποίες συμπεριλαμβάνουν μη επανδρωμένα αεροσκάφη σταθερής πτέρυγας, πολλών κινητήρων καθώς και επίγειων οχημάτων [19].

Παράλληλα, το ArduPilot διαθέτει μεγάλη βιβλιογραφία και μια υποστηρικτική διαδικτυακή κοινότητα. Αναλυτικότερα, παρέχει την δυνατότητα στους χρήστες του για την εύρεση σεμιναρίων, φόρουμ και πληροφορίων τα οποία μπορούν να συνεισφέρουν ενεργά στην ενασχόληση με τα drones [19].

Ολοκληρώνοντας, το εν λόγω λογισμικό ασχολείται περισσότερο με την διαμόρφωσή του προς τον χρήστη. Αυτό συμβαίνει καθώς διαθέτει έναν φιλικό προς τον εκάστοτε χειριστή επίγειο σταθμό ελέγχου ο οποίος ονομάζεται Mission Planner. Αυτός απλοποιεί κατά πολύ την διαδικασία διαμόρφωσης της πτήσης αλλά και τον σχεδιασμό αποστολών [19].

Σύμφωνα με τα παραπάνω, το λογισμικό το οποίο επιλέχθηκε για να χρησιμοποιεί το drone της παρούσας διπλωματικής εργασίας είναι το PX4-Autopilot. Αυτό κρίθηκε καταλληλότερο καθώς, η εφαρμογή χρειάζεται ακρίβεια και αξιοπιστία. Ωστόσο, τον πιο σημαντικό ρόλο διαδραμάτισε το γεγονός ότι υποστηρίζει εφαρμογές κατά τις οποίες δεν πραγματοποιείται χρήση GPS. Αυτό περιγράφει την διπλωματική εργασία καθώς μελετάται η πλοήγηση σε εσωτερικό χώρο όπου το GPS δεν λαμβάνει σήμα.[19]

# 4 ΟΙ ΑΙΣΘΗΤΗΡΕΣ ΤΟΥ DRONE

## 4.1 Ο αισθητήρας UWB

### 4.1.1 Γενικός ορισμός του UWB

Ο αισθητήρας UWB ορίσθηκε για πρώτη φορά από την αμερικανική FCC και μπορεί να χαρακτηριστεί με τον συνδυασμό τεσσάρων διαφορετικών εννοιών οι οποίες είναι το εύρος ζώνης, η κεντρική συχνότητα, το κλασματικό εύρος ζώνης και ο πομπός υπερύψηλού εύρους ζώνης. Ξεκινώντας, αναφέρεται το εύρος ζώνης. Αυτό είναι η ζώνη συχνοτήτων που οριοθετείται από τα σημεία που είναι 10 dB κάτω από την υψηλότερη ακτινοβολούμενη εκπομπή, με βάση το πλήρες σύστημα μετάδοσης συμπεριλαμβανομένης της κεραίας. Το ανώτερο όριο συμβολίζεται ως $f_H$ και το κατώτερο όριο αναγράφεται ως $f_L$. Η συχνότητα στην οποία η υψηλότερη ακτινοβολούμενη εκπομπή λαμβάνει χώρα χαρακτηρίζεται ως $f_M$. [20]

Συνεχίζοντας, ορίζονται μαθηματικά τα υπόλοιπα τρία μεγέθη. Πιο αναλυτικά, η κεντρική συχνότητα του συστήματος συμβολίζεται με $f_c$ και δίνεται από την σχέση:

$$f_c = \frac{f_H + f_L}{2} \qquad (4.1.1)$$

Ταυτόχρονα ως κλασματικό εύρος ζώνης δίνεται το κλάσμα:

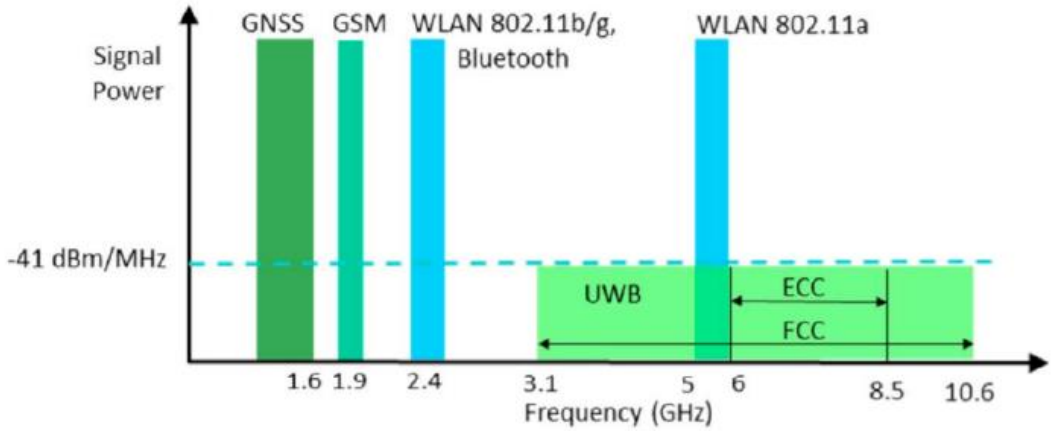
$$\frac{2 \cdot (f_H - f_L)}{f_H + f_L} \qquad (4.1.2)$$

Ολοκληρώνοντας, ως πομπός υπερύψηλου εύρους ζώνης χαρακτηρίζεται ένας ακτινοβολητής ο οποίος, σε οποιαδήποτε χρονική στιγμή, παρουσιάζει κλασματικό εύρος ζώνης ίσο ή μεγαλύτερο από 0,20 ή έχει εύρος ζώνης ίσο με ή μεγαλύτερο από 500 MHz, ανεξάρτητα από το κλασματικό εύρος ζώνης [21]. Οι εκφράσεις αυτές παρέχονται μαθηματικά από τις σχέσεις:

$$εύρος\ ζώνης \geq 500\ MHz \qquad (4.1.3)$$

$$\frac{2 \cdot (f_H - f_L)}{f_H + f_L} \geq 0{,}2 \qquad (4.1.4)$$

Το UWB πρέπει να συνυπάρχει με άλλα συστήματα, τεχνολογίες και υπηρεσίες επικοινωνίας που λειτουργούν στις ίδιες κατανεμημένες ζώνες συχνοτήτων, οπότε απαιτείται τα επίπεδα εκπομπής των σημάτων του να είναι επαρκώς χαμηλά ώστε να αποφεύγονται οι παρεμβολές. Σύμφωνα με τα Ευρωπαϊκά πρότυπα τα όρια που πρέπει να τηρούνται για τις συχνότητας είναι 3,4-5,0 GHz και 6,0-8,5 GHz, με δυνατότητα επέκτασης έως τα 9 GHz. Αυτά φαίνονται στην Εικόνα 4.1-1 όπου γίνεται και διαγραμματική σύγκριση με άλλες τεχνολογίες [22].




**Εικόνα 4.1-1: Σύγκριση τεχνολογίας UWB με άλλες παρόμοιες τεχνολογίες [21]**

### 4.1.2 Πλεονεκτήματα χρήσης της τεχνολογίας UWB

Στην παρούσα διπλωματική εργασία επιλέχθηκε να χρησιμοποιηθεί η τεχνολογία UWB εξαιτίας της ακρίβειας, της ανθεκτικότητας, της καινοτομίας και της μείωσης του κόστους που δύναται να προσφέρει. Στο παρόν υπό κεφάλαιο κρίνεται σημαντικό να σημειωθούν αναλυτικά τα πλεονεκτήματα αυτής με στόχο να αιτιολογηθεί πλήρως η επιλογή της.

Ξεκινώντας, ένα από τα κύρια πλεονεκτήματα είναι η δυνατότητα μετάδοσης δεδομένων υψηλού ρυθμού. Τα συστήματα UWB, λειτουργώντας στη συχνότητα των GHz, μπορούν να υποστηρίξουν ρυθμούς μετάδοσης που ξεπερνούν τα 500 Mb/s εντός εμβέλειας 10 μέτρων. Επιπλέον, τα συστήματα UWB διαθέτουν υψηλή ακρίβεια στην εμβέλεια, με δυνατότητα ανάλυσης χρόνου που επιτρέπει τον εντοπισμό και την τοποθέτηση με ακρίβεια κάτω του εκατοστού. Αυτά είναι χαρακτηριστικά τα οποία είναι απαραίτητα για το drone καθώς απαιτείται η δημιουργία συστήματος το οποίο θα έχει ως βασικό συστατικό του την ακρίβεια [23].

Ένα άλλο πλεονέκτημα των συστημάτων αυτών είναι η διείσδυση χαμηλών απωλειών. Αυτό επιτρέπει τη λειτουργία τόσο υπό οπτική επαφή όσο και χωρίς οπτική επαφή. Η ικανότητα αυτή είναι εξαιρετικά χρήσιμη σε εσωτερικούς χώρους και ιδιαίτερα βιομηχανικές εγκαταστάσεις όπου υπάρχει πληθώρα εμποδίων εξαιτίας των μηχανημάτων. Τα UWB συστήματα είναι επίσης ανθεκτικά στην εξασθένιση πολλαπλών διαδρομών, καθιστώντας τα ικανά να εκτελούν συνεχείς διαδρομές ακόμη και σε περιβάλλοντα που είναι σύνθετα, μειώνοντας την πολυπλοκότητα του πομποδέκτη και ενισχύοντας την απόδοση του συστήματος [23].

Τα συστήματα UWB είναι επίσης εξαιρετικά επεκτάσιμα, καθώς ο επαναπροσδιορίσιμος σχεδιασμός τους επιτρέπει την ανταλλαγή υψηλής απόδοσης δεδομένων με ευελιξία στις εφαρμογές. Τέλος, η υλοποίηση πομποδέκτη χαμηλού κόστους είναι δυνατή μέσω της ενσωμάτωσης διαφόρων τσιπ. Η απλότητα των κυκλωμάτων πομποδεκτών χαμηλής ισχύος, που δεν απαιτούν ενδιάμεσους ταλαντωτές, καθιστά το UWB μια οικονομικά αποδοτική λύση για την μετάδοση δεδομένων. Επομένως, η απλότητα και το μειωμένο κόστος των συστημάτων τα θέτουν κατάλληλα για ακαδημαϊκή χρήση [23].

### 4.1.3 Η επιλογή του τρόπου τοποθέτησης του αισθητήρα UWB

Οι συσκευές οι οποίες αποτελούν το σύστημα UWB μπορούν να διακριθούν σε δύο κατηγορίες αυτές των αγκυρών (anchor) και των ετικετών (tag). Ως άγκυρες, ορίζονται οι πομποί οι οποίοι εκπέμπουν τις γνωστές θέσεις που χρειάζονται για να καθορίσουν την θέση της ετικέτας. Παράλληλα, ως ετικέτες, σημειώνονται οι συσκευές που πρέπει να εντοπισθούν και είναι τοποθετημένες στον στόχο [21]. Στην παρούσα διπλωματική εργασία η ετικέτα βρίσκεται στην κάτω πλευρά του προς μελέτη drone καθώς ζητούμενο είναι ο εντοπισμός της θέσης του.

Σύμφωνα με πρακτικούς κανόνες θα πρέπει να ισχύει ότι ο αριθμός των αγκυρών θα πρέπει να είναι μεγαλύτερος ή ίσος με τον αριθμό των ετικετών προστιθέμενο κατά ένα. Αυτό μαθηματικά παρέχεται από την σχέση:

$$anhors \geq tags + 1 \qquad (4.3.1)$$

Ο κανόνας αυτός συνεπάγεται ότι για εύρεση θέσης σε δισδιάστατο επίπεδο χρειάζονται τουλάχιστον τρεις άγκυρες ενώ ο τρισδιάστατος χώρος απαιτεί τουλάχιστον τέσσερεις από αυτές. Ωστόσο, όσο ο αριθμός των αγκυρών αυξάνεται τόσο αυξάνει και η ακρίβεια του συστήματος. Παράλληλα, θεωρείται σημαντικό να μην παραβιαστεί το ελάχιστο όριο τοποθετημένων αγκυρών έτσι ώστε να μπορέσουν να περιοριστούν προβλήματα μη ορατότητας του συστήματος. Την ίδια στιγμή, εξίσου απαραίτητη παρατήρηση είναι και η αποφυγή τοποθέτησης όλων των αγκυρών στην ίδια ευθεία εάν πρόκειται για πρόβλημα δύο διαστάσεων ή στο ίδιο επίπεδο εάν σε αυτό εισέρχεται και η τρίτη διάσταση. [21]

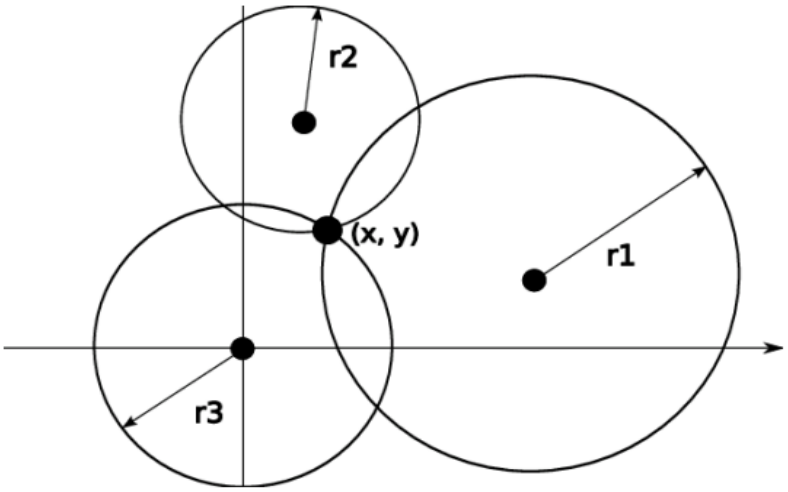
Ακολουθώντας τους παραπάνω κανόνες προτείνεται οι άγκυρες οι οποίες θα τοποθετηθούν να είναι τέσσερεις σε αριθμό. Την ίδια στιγμή, εξαιτίας του διαθέσιμου χώρου του εργαστηρίου στο οποίο θα γίνουν οι πειραματικές δοκιμές αυτά θα εγκατασταθούν σε σημεία του τοίχου υλοποιώντας ένα τετραγωνικό σχήμα. Αυτό σημαίνει ότι δεν είναι δυνατό να υπάρξει τοποθέτηση των αγκυρών σε σημαντικά διαφορετικά ύψη. Για αυτό τον λόγο, η τεχνολογία των UWB θα αξιοποιηθεί με σκοπό να υπολογίζει μόνο την δισδιάστατη θέση μεταξύ του drone και της βάσης ενώ το ύψος θα παραμένει σταθερό.

Ωστόσο, προτείνεται να χρησιμοποιηθούν τέσσερις συσκευές αντί για τρεις για να επιτρέπουν την εύρεση θέσης ακόμα και όταν μια σταματήσει να λειτουργεί αυξάνοντας έτσι την ακρίβεια εκτίμησης.

### 4.1.4 Ο τρόπος εκτίμησης της θέσης του drone

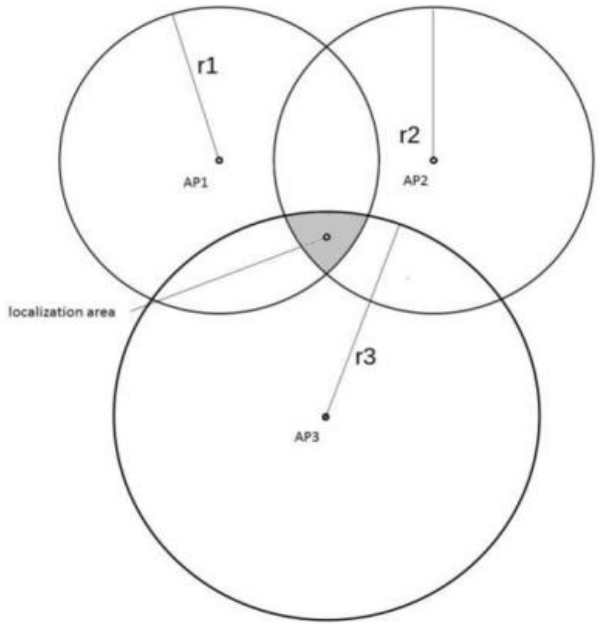
Έχοντας ολοκληρώσει την επιλογή των συσκευών και των θέσεων τους θα πρέπει να δοθεί η δυνατότητα εκτίμησης της θέσης του drone με ακρίβεια.

Η εκτίμηση αυτή ταυτίζεται με την εύρεση της ακριβής θέσης της ετικέτας από τις άγκυρες. Πραγματοποιώντας την παραδοχή ότι οι αποστάσεις δεν επηρεάζονται από σφάλματα η λύση δίνεται μαθηματικά με παρεμβολή των κύκλων των οποίων το κέντρο βρίσκεται στην θέση των αγκυρών και η ακτίνα τους ορίζεται ως η απόσταση μεταξύ αυτών και της ετικέτας. Αυτό παρουσιάζεται σχηματικά στην Εικόνα 4.1-2.




**Εικόνα 4.1-2: Οι ιδανικές αποστάσεις μεταξύ των ζητούμενων σημείων [24]**

Ωστόσο, σε όλες τις πραγματικές λύσεις οι μετρήσεις επηρεάζονται από σφάλματα με αποτέλεσμα οι κύκλοι να μην συμβάλλουν ακριβώς σε ένα σημείο αλλά σε μια περιοχή. Το φαινόμενο αυτό παρατηρείται στην Εικόνα 4.1-3.

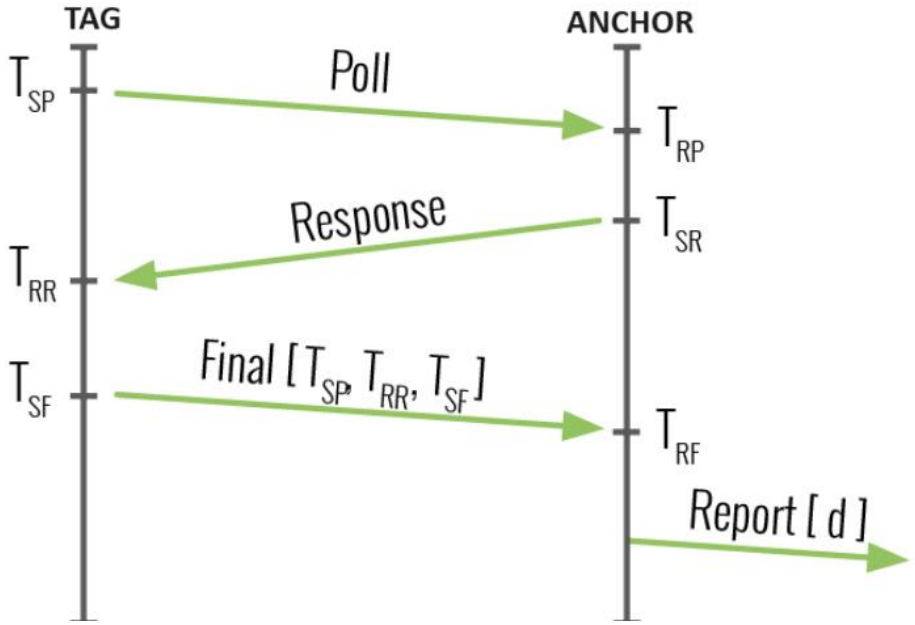



**Εικόνα 4.1-3: Οι πραγματικές αποστάσεις μεταξύ των ζητούμενων σημείων [25]**

Το UWB αξιοποιεί το χρόνο πτήσης (Time of Flight - ToF), ο οποίος είναι μια μέθοδος μέτρησης της απόστασης μεταξύ δύο ραδιοπομπών και δεκτών πολλαπλασιάζοντας το χρόνο κατά τον οποίο το σήμα ταξιδεύει με την ταχύτητα του φωτός. Αξιοποιώντας αυτή τη βασική αρχή, η τεχνολογία UWB μπορεί να εφαρμοστεί με διαφορετικούς τρόπους ανάλογα με την ανάγκη της εκάστοτε εφαρμογής. Οι επιλογές που δίνονται είναι η διαφορά χρόνου άφιξης (Time Difference of Arrival - TDoA), η αμφίδρομη εμβέλεια (Two Way Ranging) και η διαφορά φάσης άφιξης (Phase Difference of Arrival - PDoA) [26].

Ειδική αναφορά θα πραγματοποιηθεί στην μέθοδο αμφίδρομης εμβέλειας καθώς αυτή επιλέγεται να χρησιμοποιηθεί στην παρούσα διπλωματική εργασία. Πιο συγκεκριμένα, στην Εικόνα 4.1-4 παρατηρείται σχηματικά η εν λόγω μέθοδος. Αναλύοντας την, αυτή καθορίζει τον χρόνο πτήσης του σήματος ο οποίος συμβολίζεται με «$T_{RF}$» και στην συνέχεια υπολογίζει την απόσταση μεταξύ των κόμβων πολλαπλασιάζοντας τον χρόνο αυτό με την ταχύτητα του φωτός. Ωστόσο, είναι χρήσιμο να σημειωθεί ότι η διαδικασία αυτή εφαρμόζεται μεταξύ της ετικέτας και μίας μόνο άγκυρας σε ένα δεδομένο χρονικό διάστημα. [27]

Εισάγοντας περισσότερες λεπτομέρειες, για την ολοκλήρωση της μέτρησης της απόστασης πρέπει να γίνει ανταλλαγή τριών μηνυμάτων. Αρχικά, η ετικέτα αρχικοποιεί την διαδικασία στέλνοντας το μήνυμα με όνομα «Poll» στην γνωστή διεύθυνση της άγκυρας. Αυτό γίνεται σε χρόνο που σημειώνεται ως «$T_{SP}$» (Time of Sending Poll). Η άγκυρα καταγράφει το χρόνο λήψης του μηνύματος με όνομα «Poll» σε χρόνο που σημειώνεται ως «$T_{RP}$» και απαντά με το ανάλογο μήνυμα στο χρόνο «$T_{SR}$». Έτσι, η ετικέτα κατά τη λήψη του μηνύματος απάντησης καταγράφει τη χρονική στιγμή η οποία συμβολίζεται με $T_{RR}$ και συνθέτει το τελικό μήνυμα, όπου περιλαμβάνονται οι πληροφορίες για την ταυτότητα του μηνύματος καθώς και τα «$T_{SP}$», «$T_{RR}$» και «$T_{SF}$», που κρίνονται απαραίτητα για τους τελικούς υπολογισμούς. Με βάση το χρόνο λήψης του τελικού μηνύματος «$T_{RF}$» αλλά και τις πληροφορίες που παρέχονται από αυτό, η άγκυρα μπορεί να εντοπίσει το χρόνο πτήσης του σήματος UWB. [27]




**Εικόνα 4.1-4: Μέθοδος αμφίδρομης εμβέλειας [27]**

Όλα όσα αναφέρθηκαν παραπάνω συνοψίζονται μαθηματικά στις ακόλουθες σχέσεις:

$$απόσταση = ToF \cdot ταχύτητα\ του\ φωτός \quad (4.4.1)$$

$$ToF = \frac{[(T_{RR} - T_{SP}) - (T_{SR} - T_{RP}) + (T_{RF} - T_{SR}) - (T_{SF} - T_{RR})]}{4} \quad (4.4.2)$$

Με στόχο την διόρθωση του ζητήματος της ακρίβειας και την εύρεση μιας ευσταθούς λύσης στο σύστημα θα πρέπει να χρησιμοποιηθεί μια μαθηματική μέθοδος η οποία θα διορθώνει το σφάλμα μεταξύ της εκτιμώμενης και της πραγματικής θέσης. Στην παρούσα διπλωματική εργασία επιλέγεται να χρησιμοποιηθεί η λύση των γραμμικών ελαχίστων τετραγώνων( Linear Least Squares).

Προχωρώντας στην ανάλυση της μεθόδου ορίζονται αρχικά το διάνυσμα ( $\widehat{x_{tag}}, \widehat{y_{tag}}, \widehat{z_{tag}}$ ) το οποίο προσδιορίζει την εκτιμώμενη θέση της ετικέτας καθώς και τα σημεία τα οποία βρίσκονται οι άγκυρες. Αυτά είναι τα $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$, $P_3(x_3, y_3, z_3)$ και $P_4(x_4, y_4, z_4)$. Γνωρίζοντας αυτά υπολογίζονται οι αποστάσεις της ετικέτας από κάθε άγκυρα με βάση τον τύπο της Ευκλείδειας απόστασης. Αυτές συμβολίζονται με $s_i$ με i= 1,2,3,4. Επομένως λαμβάνεται,

$$\begin{cases} (\widehat{x_{tag}} - x_1)^2 + (\widehat{y_{tag}} - y_1)^2 + (\widehat{z_{tag}} - z_1)^2 = s_1^2 \\ (\widehat{x_{tag}} - x_2)^2 + (\widehat{y_{tag}} - y_2)^2 + (\widehat{z_{tag}} - z_2)^2 = s_2^2 \\ (\widehat{x_{tag}} - x_3)^2 + (\widehat{y_{tag}} - y_3)^2 + (\widehat{z_{tag}} - z_3)^2 = s_3^2 \\ (\widehat{x_{tag}} - x_4)^2 + (\widehat{y_{tag}} - y_4)^2 + (\widehat{z_{tag}} - z_4)^2 = s_4^2 \end{cases} \quad (4.4.3)$$

Αναλύοντας την σχέση (2), απομονώνοντας τις συντεταγμένες της ετικέτας και σχηματίζοντας μητρωική μορφή λαμβάνεται ότι:

$$\begin{bmatrix} 1 & -2x_1 & -2y_1 & -2z_1 \\ 1 & -2x_2 & -2y_2 & -2z_2 \\ 1 & -2x_3 & -2y_3 & -2z_3 \\ 1 & -2x_4 & -2y_4 & -2z_4 \end{bmatrix} \begin{bmatrix} \widehat{x_{tag}}^2 + \widehat{y_{tag}}^2 + \widehat{z_{tag}}^2 \\ \widehat{x_{tag}} \\ \widehat{y_{tag}} \\ \widehat{z_{tag}} \end{bmatrix} = \begin{bmatrix} s_1^2 - x_1^2 - y_1^2 - z_1^2 \\ s_2^2 - x_2^2 - y_2^2 - z_2^2 \\ s_3^2 - x_3^2 - y_3^2 - z_3^2 \\ s_4^2 - x_4^2 - y_4^2 - z_4^2 \end{bmatrix} \quad (4.4.4)$$

Αυτή μπορεί συνοπτικά να γραφεί με την μορφή :

$$A \cdot \hat{x} = b \quad (4.4.5)$$

όπου $A = \begin{bmatrix} 1 & -2x_1 & -2y_1 & -2z_1 \\ 1 & -2x_2 & -2y_2 & -2z_2 \\ 1 & -2x_3 & -2y_3 & -2z_3 \\ 1 & -2x_4 & -2y_4 & -2z_4 \end{bmatrix}$, $\hat{x} = \begin{bmatrix} \widehat{x_{tag}}^2 + \widehat{y_{tag}}^2 + \widehat{z_{tag}}^2 \\ \widehat{x_{tag}} \\ \widehat{y_{tag}} \\ \widehat{z_{tag}} \end{bmatrix}$ και $b = \begin{bmatrix} s_1^2 - x_1^2 - y_1^2 - z_1^2 \\ s_2^2 - x_2^2 - y_2^2 - z_2^2 \\ s_3^2 - x_3^2 - y_3^2 - z_3^2 \\ s_4^2 - x_4^2 - y_4^2 - z_4^2 \end{bmatrix}$

Η λύση της εξίσωσης (4.4.3) δίνεται από την σχέση:

$$\hat{x} = (A^T A)^{-1} A^T b \quad (4.4.6)$$

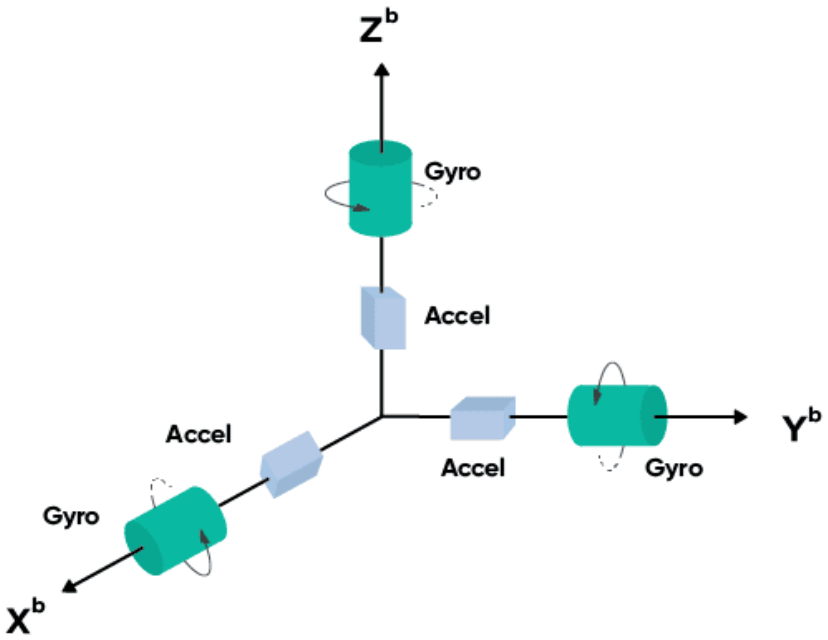
Σε αυτό το σημείο αξίζει να σημειωθεί ότι όλα τα στοιχεία του πίνακα $(A^T A)^{-1} A^T$ αναφέρονται μόνο στις συντεταγμένες των σημείων που έχουν ορισθεί ως άγκυρες. Σε αντίθεση με αυτό, το διάνυσμα b αποτελείται από τις αποστάσεις μεταξύ του αγνώστου σημείου, που είναι η ετικέτα, και όλων των σημείων αναφοράς.

Η επιλογή αυτού του αλγορίθμου έναντι των υπολοίπων έχει γίνει εξαιτίας του χαμηλού υπολογιστικού του κόστους καθώς και της σταθερότητας που προσφέρει. Αναλύοντας το πρώτο, η μόνη πράξη η οποία καταναλώνει υπολογιστική ισχύ σε επίπεδο κώδικα είναι η αντιστροφή πινάκων που απαιτείται για την εύρεση του τελικού αποτελέσματος. Με άλλα λόγια, πρόκειται για την πράξη $(A^T A)^{-1}$. Συνεχίζοντας με το δεύτερο, ο αλγόριθμος προσφέρει σταθερότητα κατά τις διεργασίες του καθώς τα αρχικά σημεία με τα οποία θα εκκινήσει παρέχονται από τον χρήστη. Ωστόσο, το βασικό του μειονέκτημα είναι η έλλειψη ακρίβειας συγκριτικά με τους υπόλοιπους αλγορίθμους. Αυτό γίνεται καθώς υλοποιεί γραμμικοποίηση του συστήματος [21].

## 4.2 Το γυροσκόπιο (IMU)

### 4.2.1 Γενικός ορισμός του IMU

Το γυροσκόπιο (IMU) είναι μια συσκευή, είτε ηλεκτρομηχανική είτε στερεάς κατάστασης, που περιέχει μια σειρά αισθητήρων ικανών να μετρήσουν την κίνηση ενός αντικειμένου. Συγκεκριμένα, το IMU ανιχνεύει τη γραμμική επιτάχυνση η οποία είναι ο ρυθμός μεταβολής της ταχύτητας και τον γωνιακό ρυθμό δηλαδή την μεταβολή της γωνιακής ταχύτητας γύρω από τους άξονες X, Y και Z. Αυτή η διαδικασία επιτρέπει την κατανόηση του προσανατολισμού και της κίνησης του αντικειμένου. Την ίδια στιγμή, αυτός ο αισθητήρας μετατρέπει την ανιχνευόμενη αδράνεια, δηλαδή τις δυνάμεις που δημιουργούνται λόγω της αντίστασης ενός αντικειμένου στην αλλαγή κατεύθυνσης, σε δεδομένα εξόδου που περιγράφουν την κίνηση του αντικειμένου. Αυτά τα δεδομένα χρησιμοποιούνται από διάφορα συστήματα, όπως για τον έλεγχο ενός οχήματος όπου στην περίπτωση της παρούσας διπλωματικής εργασίας είναι το μη επανδρωμένο εναέριο όχημα του εργαστηρίου. Τέλος, περιγράφοντας με περισσότερη λεπτομέρεια την έξοδο αυτής της συσκευής περιλαμβάνει συνήθως δεδομένα από επιταχυνσιόμετρα, που μετρούν τη γραμμική επιτάχυνση κατά μήκος κάθε άξονα και γυροσκόπια, που μετρούν την ταχύτητα περιστροφής ή γωνιακή ταχύτητα γύρω από κάθε άξονα. Αυτά μαζί με τους άξονες στους οποίους ενεργούν φαίνονται στην Εικόνα 4.2-1. [28]




**Εικόνα 4.2-1: Επιταχυνσιόμετρα και γυροσκόπια στους άξονες X, Y,Z [28]**

### 4.2.2 Πλεονεκτήματα και μειονεκτήματα της συσκευής IMU

Η συσκευή IMU διαθέτει μια σειρά πλεονεκτημάτων που την καθιστά ιδιαίτερα χρήσιμη σε πληθώρα εφαρμογών.

Πρώτον, είναι μικρή, ελαφριά και σχετικά φθηνή. Πιο αναλυτικά, το IMU είναι συνήθως πολύ μικρότερη και ελαφρύτερη συσκευή από άλλα συστήματα πλοήγησης, γεγονός που τα καθιστά ιδανικά για εφαρμογές που απαιτούν μείωση του μεγέθους και του βάρους, όπως η ρομποτική [29].

Επιπλέον, δύναται να χρησιμοποιηθεί σε διάφορα περιβάλλοντα. Ειδικότερα, το IMU μπορεί να λειτουργεί αποτελεσματικά ακόμα και σε περιβάλλοντα όπου το GPS δεν είναι διαθέσιμο, όπως κάθε είδους εσωτερικού χώρου, καθιστώντας τα ιδανικά για πλοήγηση σε σχεδόν όλες τις περιοχές [29].

Τέλος, το IMU έχει την ικανότητα να παρέχει συνεχείς μετρήσεις, ακόμη και όταν το αντικείμενο κινείται γρήγορα. Αυτή η δυνατότητα συνεχούς παρακολούθησης της κίνησης είναι εξαιρετικά σημαντική για εφαρμογές όπως αυτές των μη επανδρωμένων εναέριων οχημάτων κατά τις οποίες απαιτείται διαρκώς εντοπισμός του συστήματος [29].

Από τα παραπάνω η συσκευή IMU εμφανίζει μια σειρά πλεονεκτημάτων τα οποία οδηγούν στην επιλογή της για χρήση στο προς κατασκευή drone. Ωστόσο, υπάρχουν και μειονεκτήματα τα οποία πρέπει να ληφθούν υπόψη κατά την χρήση.

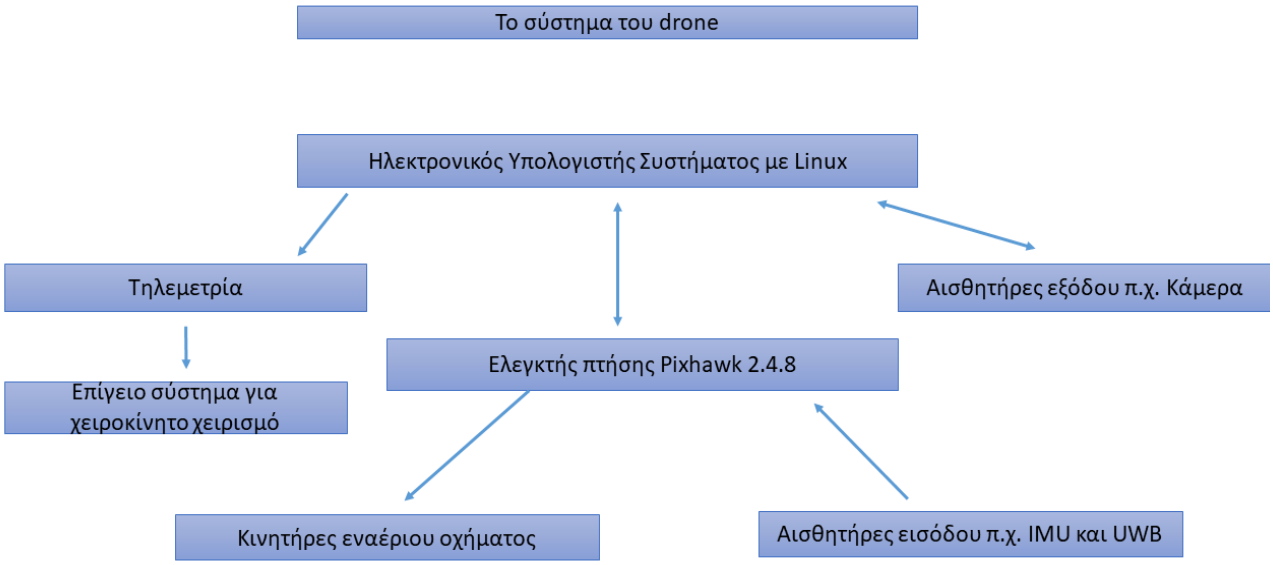
Ξεκινώντας, η ακρίβεια των μετρήσεων που παρέχει η IMU μπορεί να ελαττωθεί με την πάροδο του χρόνου λόγω ολίσθησης. Αυτό σημαίνει ότι μπορεί σταδιακά να χάνει την ικανότητά της να μετρά με ακρίβεια τον προσανατολισμό, την ταχύτητα και την επιτάχυνση του αντικειμένου, οδηγώντας σε μειωμένη αξιοπιστία των δεδομένων [29].

Παράλληλα, η συσκευή αυτή εμφανίζει ευαισθησία στο θόρυβο. Αυτό σημαίνει ότι οι μετρήσεις που παρέχουν μπορούν να επηρεαστούν από εξωτερικούς παράγοντες όπως οι δονήσεις, οι ηλεκτρομαγνητικές παρεμβολές και οι μεταβολές της θερμοκρασίας, με αποτέλεσμα την αλλοίωση των δεδομένων [29].

Τέλος, το IMU απαιτεί συχνά βαθμονόμηση για να διασφαλιστεί η ακρίβεια των μετρήσεων. Αυτό αποτελεί πρόβλημα καθώς η διαδικασία της βαθμονόμησης μπορεί να είναι χρονοβόρα και πολύπλοκη, απαιτώντας τεχνογνωσία [29].

# 5 Ο ΚΩΔΙΚΑΣ ΠΛΟΗΓΗΣΗΣ

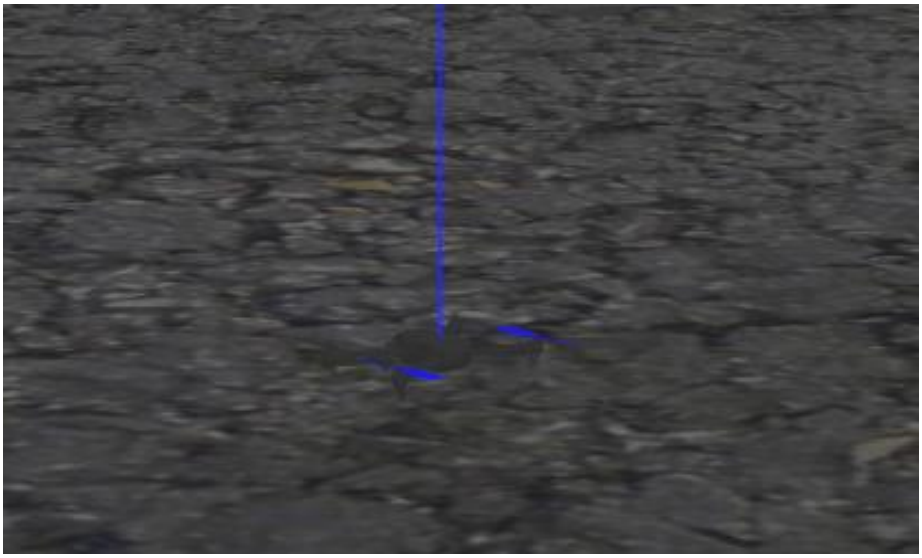
Το παρόν κεφάλαιο έχει ως στόχο την κατανόηση του κώδικα που έχει υλοποιηθεί/ χρησιμοποιηθεί με σκοπό την διεκπεραίωση της εν λόγω διπλωματικής εργασίας. Αυτός προσπαθεί να προσομοιώσει με μαθηματικό τρόπο την λειτουργία των αισθητήρων UWB και IMU. Πιο συγκεκριμένα, έμφαση δίνεται στον πρώτο από αυτούς καθώς πρόκειται για τον βασικό αισθητήρα σύμφωνα με τον οποίο υπολογίζεται η θέση του μη επανδρωμένου εναέριου οχήματος κάθε χρονική στιγμή. Ο κώδικας για αυτόν δεν προϋπήρχε και κατασκευάσθηκε στα πλαίσια της εν λόγω εργασίας. Στη συνέχεια, θεωρείται χρήσιμο να αποδοθεί διάγραμμα το οποίο καταδεικνύει όλα τα πεδία τα οποία είναι απαραίτητα για την πτήση ενός drone. Αυτό φαίνεται στην Εικόνα 5-1. Η παρέμβαση γίνεται στο τελικό πλαίσιο των αισθητήρων οι οποίοι είναι υπεύθυνοι για να παρέχουν πληροφορίες για την κατάσταση του drone κάθε χρονική στιγμή.




**Εικόνα 5-1: Συνολικό διάγραμμα πεδίων του drone**

## 5.1 Το drone Iris της εταιρίας PX4

Ο κώδικας ο οποίος πραγματοποιήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας έχει ως στόχο του την πλοήγηση ενός drone το οποίο είναι διαθέσιμο για προσομοιώσεις από την εταιρία PX4.Πιο συγκεκριμένα, αυτό έχει το όνομα iris, είναι κατασκευασμένο με την λογική του τετρακοπτέρου, και είναι σε σχήμα Χ. Για καλύτερη κατανόηση της κατασκευής του αυτό φαίνεται στην Εικόνα 5.1-1.



**Εικόνα 5.1-1: Το drone Iris**

Παράλληλα, η εταιρία διαθέτει το αρχείο της μορφής sdf το οποίο ορίζει τα χαρακτηριστικά του. Αναλυτικότερα, πρόκειται για ένα μικρό drone το οποίο ζυγίζει 1,5 κιλά. Την ίδια στιγμή οι διαστάσεις του βασικού του σκελετού σημειώνονται να είναι 470 x 470 x 110 mm. Ταυτόχρονα, οι ενδείξεις των ροπών αδράνειας καταδεικνύουν ότι είναι συμμετρικό ως προς τους κύριους άξονες X, Y, Z.

Όλα τα παραπάνω χαρακτηριστικά οδηγούν στο συμπέρασμα ότι το drone iris έχει πολλά κοινά με το drone που έχει κατασκευαστεί στο εργαστήριο Τεχνολογίας των Κατεργασιών της Σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου. Αυτό συμβαίνει καθώς και τα δύο μη επανδρωμένα εναέρια οχήματα έχουν παρόμοιες διαστάσεις, χαρακτηρίζονται ως μικρού μεγέθους αλλά και διαθέτουν το ίδιο σχήμα και φιλοσοφία καθώς πρόκειται για τετρακόπτερα σχήματος Χ. Τα προαναφερόμενα κοινά χαρακτηριστικά είναι το βασικό πλεονέκτημα το οποίο οδηγεί στην επιλογή του ως drone χρήσιμο για τις προσομοιώσεις.

## 5.2 Ο κώδικας της πλοήγησης του συστήματος

Στο παρόν υπό κεφάλαιο πρόκειται να πραγματοποιηθεί εξήγηση των βασικών σημείων του κώδικα ο οποίος υλοποιήθηκε/χρησιμοποίηθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας. Ο κώδικας αυτός έχει γραφεί σε γλώσσα προγραμματισμού C++. Ειδικότερα, αρχικά θα διευκρινιστεί η λειτουργία όλων των συναρτήσεων που δημιουργήθηκαν και στην συνέχεια θα δοθεί μια ανάλυση για την λειτουργία της βασικής συνάρτησης (main function).

### 5.2.1 Εξήγηση των επιμέρους συναρτήσεων

Με στόχο την καλύτερη κατανόηση του κώδικα που δημιουργήθηκε θεωρείται πρέπον να αναλυθούν οι συναρτήσεις που τον αποτελούν. Συγκεκριμένα, η πρώτη συνάρτηση που θα περιγραφεί ονομάζεται imuCallback και ο κώδικάς της αποδίδεται παρκάτω. Η συνάρτηση αυτή έχει ως όρισμα τα δεδομένα τα οποία προέρχονται από τον αισθητήρα του γυροσκοπίου και είναι υπεύθυνη για την αποθήκευση τους σε μεταβλητές έτσι ώστε να μπορούν να δημοσιευθούν στα κατάλληλα θέματα.

```
void imuCallback(const sensor_msgs::Imu::ConstPtr& msg) {

  acc_data = msg->linear_acceleration; //message for linear acceletation

  vel_data = msg->angular_velocity; // message for angular velocity

  position_data = msg->orientation; // message for orientation

}
```

Συνεχίζοντας, η επόμενη συνάρτηση που παρατηρείται ονομάζεται state_cb. Αυτή γράφτηκε με σκοπό να λαμβάνει δεδομένα που αντιπροσωπεύουν την κατάσταση του drone και να ενημερώνει για την τρέχουσα κατάσταση αυτού με βάση τα προηγούμενα δεδομένα. Ο κώδικας που την δημιουργεί αναγράφεται στην συνέχεια με στόχο την ακριβή κατανόηση.

```
void state_cb(const mavros_msgs::State::ConstPtr& msg) {

   current_state = *msg;

}
```

Προχωρώντας στην ανάλυση του κώδικα εντοπίζεται άλλη μια συνάρτηση η οποία έχει γραφτεί για την εξαγωγή δεδομένων από τον προσομοιωμένο αισθητήρα UWB. Η συνάρτηση αυτή ονομάζεται UWBcallback και λαμβάνει ως όρισμα τα μηνύματα τα οποία προέρχονται από το Model States του Gazebo το οποίο περιλαμβάνει τη θέση και την ταχύτητα του drone μεγέθη τα οποία ταυτίζονται με αυτά της ετικέτας. Σκοπός της συνάρτησης είναι να υπολογίζει κάθε χρονική στιγμή τις αποστάσεις της ετικέτας από τις γνωστές άγκυρες μέσω της χρήσης της συνάρτησης robotposition η οποία θα αναλυθεί στη συνέχεια του παρόντος κεφαλαίου. Έπειτα, έχοντας την γνώση των αποστάσεων αυτών και χρησιμοποιώντας μια επιπλέον συνάρτηση η οποία θα αναφερθεί σε αυτό το κεφάλαιο και καλείται position_calculation εκτιμάται η τελική θέση του drone. Ωστόσο, αυτή δεν μπορεί να δημοσιευτεί με αυτήν την μορφή σε κάποιο θέμα. Εξαιτίας αυτού δημιουργείται ένα σημείο το οποίο αποθηκεύει τις συντεταγμένες της θέσης και είναι εκείνο που τελικά δημοσιεύεται στο θέμα. Ο κώδικας της συνάρτησης που έχει αναλυθεί αποδίδεται παρακάτω.

```
void UWBcallback(const gazebo_msgs::ModelStates::ConstPtr& msg){

     //  locali_pose = msg->pose[index]; Do not need it

    //   locali_twist = msg->twist[index]; Do not need it

        Eigen::VectorXd distances = robotposition(msg);  // Calculates the distances between the tag and the anchors

      Eigen :: Vector3d robot_pos = position_calculation(anchors,distances); // Calculates the final position of the UAV

      // std::cout << "Robot: " << robot_pos.transpose() << std::endl;  MESSAGE FOR DEBUGGING


      // Store the positions

   geometry_msgs::Point pos_msg;

   pos_msg.x = robot_pos[0];

   pos_msg.y = robot_pos[1];

   pos_msg.z = robot_pos[2];

   //Make a publisher to publish the data to a topic

 pos_pub.publish(pos_msg);

}
```

Έχοντας ολοκληρώσει με την ανάλυση των συναρτήσεων οι οποίες είναι υπεύθυνες για την λήψη, κατανόηση και επεξεργασία των μηνυμάτων που προέρχονται από τους αισθητήρες πρέπει να αναλυθούν αυτές οι

οποίες προσομοιώνουν την λειτουργία του βασικού αισθητήρα UWB. Αυτές είναι συνολικά τρεις και λαμβάνουν τα ονόματα computeDistance, robotposition και position_calculation.

Ξεκινώντας με την συνάρτηση computeDistance, αυτή λαμβάνει ως όρισμα δύο μεγέθη τα οποία είναι ένα τρισδιάστατο διάνυσμα το οποίο πρέπει να περιγράφει την θέση της ετικέτας και έναν ακέραιο αριθμό ο οποίος δείχνει την άγκυρα. Στη συνέχεια, η συνάρτηση χρησιμοποιώντας αυτόν τον ακέραιο αριθμό εντοπίζει τις πλήρεις συντεταγμένες της γνωστής άγκυρας και αξιοποιώντας την μαθηματική νόρμα βρίσκει την απόσταση μεταξύ των δύο αισθητήρων. Την ίδια στιγμή, αυτή έχει ως σκοπό να προσομοιώσει με τον καλύτερο δυνατό τρόπο την πληροφορία που παρέχει η διαδικασία του χρόνου πτήσης (ToF) στο σύστημα. Ο κώδικας ο οποίος την περιγράφει αναγράφεται παρακάτω.

```
double computeDistance(const Eigen:: Vector3d& tag_position, int anchor_index) {
   Eigen:: Vector3d anchor_position = anchors.row(anchor_index);
   return (tag_position - anchor_position).norm();
}
```

Έπειτα, αναλύεται η συνάρτηση με όνομα robotposition. Η συνάρτηση αυτή χρησιμοποιεί ως όρισμα τα μηνύματα τα οποία δέχεται το σύστημα από το Model States του Gazebo. Η λογική της ξεκινά με την εύρεση της θέσης της ετικέτας η οποία είναι προσαρμοσμένο πάνω στο drone. Όταν η θέση αυτή βρεθεί καταχωρείται σε ένα τρισδιάστατο διάνυσμα και χρησιμοποιείται έτσι ώστε να υπολογισθούν οι αποστάσεις από τις άγκυρες. Αυτό γίνεται μέσω της συνάρτησης που εξηγήθηκε προηγουμένως με όνομα computeDistance και η αποθήκευση πραγματοποιείται σε ένα νέο διάνυσμα το οποίο είναι και εκείνο που παρέχεται ως έξοδος. Ο κώδικας ο οποίος αντιπροσωπεύει την παραπάνω λογική παρατίθεται παρακάτω.

```
Eigen :: VectorXd robotposition(const gazebo_msgs::ModelStates::ConstPtr& msg) {

Eigen:: VectorXd  dist(anchors.rows());
 dist.setZero(); // Initialize with zeros
   // Find the index of the tag in the model states
   std::vector<std::string>::const_iterator it = std::find(msg->name.begin(), msg->name.end(), "iris");
   if (it != msg->name.end()) {
      int tag_index = std::distance(msg->name.begin(), it);
      // Extract tag position
       Eigen ::Vector3d tag_position(msg->pose[tag_index].position.x, msg->pose[tag_index].position.y, msg-
>pose[tag_index].position.z);
      // Use of debugging message
     // std::cout << "Tag Position: " << tag_position.transpose() << std::endl;

      // Compute distance from each anchor
      for (int i = 0; i < anchors.rows(); ++i) {
         double distance = computeDistance(tag_position, i);
```

```
        dist(i) = distance;
        //Use of debugging message
    // std::cout << "Distance to anchor " << i << ": " << distance << std::endl;
      } }else {
      std::cerr << "'iris' not found in model states." << std::endl;
   }
   return dist;
   }
```

Η περιγραφή των συναρτήσεων ολοκληρώνεται με την ανάλυση της συνάρτησης με όνομα position_calculation. Αυτή είναι υπεύθυνη για την εισαγωγή της λογικής των Γραμμικών Ελαχίστων Τετραγώνων η οποία έχει δοθεί αναλυτικά στο τέταρτο κεφάλαιο της παρούσας διπλωματικής εργασίας. Η συνάρτηση προσπαθεί να υλοποιήσει σταδιακά τις μαθηματικές σχέσεις του τέταρτου κεφαλαίου και να βοηθήσει τον συνολικό κώδικα να βελτιώσει τα αποτελέσματά του. Η λογική που την διέπει φαίνεται παρακάτω σε μορφή κώδικα.

```
Eigen :: Vector3d position_calculation(const Eigen::MatrixXd& anchors, const Eigen::VectorXd& dist) {

   Eigen :: MatrixXd A = (-2 * anchors);
    int vertical = anchors.cols();
    int horizontal = anchors.rows();
    //For Debugging
   //  std::cout << "Anchors: \n" << anchors << std::endl;
  // std::cout << "Dist: \n" << dist << std::endl;
  //  ROS_INFO("Anchors (rows, cols): (%ld, %ld)", anchors.rows(), anchors.cols());
  // ROS_INFO("Dist size: %ld", dist.size());

   Eigen :: MatrixXd B = Eigen :: MatrixXd::Ones(horizontal, 1);
  //  A.conservativeResize(Eigen ::NoChange, A.cols()+1);
  // You have to make the table to have for columns with the fist one to have zeros
    Eigen:: MatrixXd Anew(4,4);
    Anew.col(0)=B;
    Anew.col(1) = A.col(0);
    Anew.col(2) = A.col(1);
```

```
   Anew.col(3) =A.col(2);

// For Debugging
 //ROS_INFO("A (rows, cols): (%ld, %ld)", A.rows(), A.cols());
//std::cout << "A: \n" << Anew << std::endl;
  Eigen :: VectorXd R = Eigen::VectorXd::Zero(vertical); // I do not use it
  Eigen :: MatrixXd b = Eigen::MatrixXd::Zero(horizontal,1);
  //I make the b matrix the way theory shows
 for (int i = 0; i < horizontal; ++i) {

    b(i,0) = (dist(i)*dist(i)) - (anchors(i,0)*anchors(i,0))- (anchors(i,1)*anchors(i,1))- (anchors(i,2)*anchors(i,2));

  }

     // Debugging message
// ROS_INFO("b size: %ld", b.size());

  Eigen:: MatrixXd Atr = Anew.transpose(); // Find the A^T
  Eigen::MatrixXd AtrA(4, 4); // Declare a new matrix

  // Multiply the tables. It was not okay to just use the * sumbol
   for (int i = 0; i < Atr.rows(); ++i) {
     for (int j = 0; j < Anew.cols(); ++j) {
        AtrA(i, j) = 0; // Initialize result element at (i, j)

        for (int k = 0; k < Atr.cols(); ++k) {
           AtrA(i, j) += Atr(i, k) * Anew(k, j);
        }
     }
  }
```

```
// For debugging. It is important to know if the table is inverible

  //Eigen:: MatrixXd AtrA =Atr* A;
  // Check if AtrA is invertible
   //if (AtrA.determinant() == 0) {
   //    ROS_ERROR("Matrix AtrA is not invertible!");
   //    return Eigen::Vector3d::Zero();
  //  }

  // std::cout << "Matrix A:\n" << A << std::endl;
  // std::cout << "Matrix A^T:\n" << Atr << std::endl;
  // std::cout << "Matrix A^T * A:\n" << AtrA << std::endl;
   //std::cout << "Determinant of A^T * A: " << det << std::endl;

  // Do the inverse
    Eigen::CompleteOrthogonalDecomposition<Eigen::MatrixXd> codA(A);
    //Find the pseudoinverse
    Eigen::MatrixXd pseudoAtrA = codA.pseudoInverse();

    //Use of debugging messages
    // std::cout << "Matrix pseudo:\n" << pseudoAtrA << std::endl;
   // Eigen:: MatrixXd Atotal = pseudoAtrA *Anew.transpose();

    Eigen:: MatrixXd yhat = pseudoAtrA*b;
    //Again a debugging message
  // std::cout << "Matrix YHAT:\n" << yhat << std::endl;

  Eigen::Vector3d x = yhat.block<3, 1>(yhat.size() - 3, 0);  // Extract last 3 elements
 //std:: cout << "x:\n"<< x << std:: endl;
 // I want positive values
      if (x[0] <0){
```

```
        x[0] = -x[0];


    }
    if(x[1]<0) {
        x[1]=-x[1];
    }


   // Try to debug
//std:: cout << "x:\n"<< x << std:: endl;


    return x;


}
```

### 5.2.2 Εξήγηση της βασικής συνάρτησης (main function)

Με στόχο την λειτουργία των συναρτήσεων που δημιουργήθηκαν και περιεγράφηκαν στο προηγούμενο κεφάλαιο πρέπει να δομηθεί με σωστό και πλήρη τρόπο η βασική συνάρτηση του κώδικα. Αυτή είναι η συνάρτηση main και είναι χαρακτηριστική της γλώσσας προγραμματισμού C++.

Ξεκινώντας, την διαδικασία κατασκευής θεωρείται σωστό να γίνει ο προσδιορισμός και η αρχικοποίηση των μεγεθών που καθορίζουν τον κώδικα. Πιο συγκεκριμένα, αυτά είναι η δημιουργία του κόμβου ο οποίος θα χρησιμοποιείται καθώς και ο καθορισμός των θέσεων των συσκευών που θα διαδραματίζουν τον ρόλο των αγκυρών. Οι θέσεις αυτές πρέπει να είναι γνωστές για το σύστημα έτσι ώστε να μπορούν να εκτελεστούν οι λειτουργίες των συναρτήσεων. Ο κώδικας ο οποίος περιγράφει την αρχικοποίηση τόσο του κόμβου όσο και των αγκυρών αποδίδεται παρακάτω.

```
   //Create and define the node
    ros::init(argc, argv, "offb_node");
    ros::NodeHandle nh;


// It was a try to initialize the anchors through fuction. It is not used
// Define anchors with their coordinates and dist;
//Eigen::MatrixXd anchors = initializeAnchors();


// Initialize the anchros
// I use four anchors in the same height
```

```
//Eigen::MatrixXd anchors(4, 3);
   anchors<<   10.0, 10.0, 3.0,
            10.0, -10.0, 3.0,
            -10.0, 10.0, 3.0 ,
            -10.0, -10.0, 3.0;
```

Συνεχίζοντας την συγγραφή της βασικής συνάρτησης, είναι σημαντικό να συμπεριληφθούν μια σειρά από καταγραφείς και εκδότες μηνυμάτων (subscribers and publishers). Σκοπός των πρώτων είναι να λαμβάνουν τα μηνύματα τα οποία παρέχει το πρόγραμμα ROS, να τα επεξεργάζονται με την κατάλληλη συνάρτηση και τέλος να τα αποθηκεύουν σε ένα θέμα. Έπειτα, οι εκδότες έχουν ως υποχρέωση να λαμβάνουν τα μηνύματα αυτά και να τα τοποθετούν σε κατάλληλα θέματα τα οποία θα επιτρέπουν την παρουσίασή τους σε διαγράμματα έτσι ώστε ο χρήστης να μπορεί να παρακολουθεί την κατάσταση του ρομποτικού συστήματος. Παράλληλα, σημαντική είναι και η τοποθέτηση δύο εντολών με το όνομα ServiceClient. Αυτές έχουν ως σκοπό το κάλεσμα δύο υπηρεσιών του MAVROS οι οποίες αναφέρονται αφενός στο όπλισμα του drone και αφετέρου στην αλλαγή της κατάστασης του η οποία καθορίζει την συμπεριφορά του. Οι καταστάσεις οι οποίες χρησιμοποιούνται συχνά είναι η αυτόματη( Auto), η σταθερή( Stabilize) και η υπό καθοδήγηση( Guided). Την ίδια στιγμή, το σημείο του κώδικα το οποίο δείχνει τις προαναφερόμενες λειτουργίες φαίνεται παρακάτω.

```
// All the publishers and subscribes


   ros::Subscriber state_sub = nh.subscribe<mavros_msgs::State>
         ("mavros/state", 10, state_cb); // Subscribe the state of the drone


   ros::Publisher local_pos_pub = nh.advertise<geometry_msgs::PoseStamped>
         ("mavros/setpoint_position/local", 10); // Publish the local position of the drone


   ros::ServiceClient arming_client = nh.serviceClient<mavros_msgs::CommandBool>
         ("mavros/cmd/arming"); //Show the arming situation


   ros::ServiceClient set_mode_client = nh.serviceClient<mavros_msgs::SetMode>
         ("mavros/set_mode"); //Show the mode of the drone


      // For the IMU system
      // Make a subscriber that collects the information from the IMU plugin
   ros::Subscriber imu_sub = nh.subscribe<sensor_msgs::Imu>("/mavros/imu/data", 10, imuCallback);
```

```
// Make publishers to receive the data and publish them in different topics
ros::Publisher imu_pub = nh.advertise<geometry_msgs::Vector3>("sofia/imu/linearaccel", 10);
ros::Publisher imu_pub1 = nh.advertise<geometry_msgs::Vector3>("sofia/imu/velocity", 10);
ros::Publisher imu_pub2 = nh.advertise<geometry_msgs::Quaternion>("sofia/imu/orientation", 10);

// For the UWB system. The publisher that prints the position of the drone
pos_pub = nh.advertise<geometry_msgs::Point>("robot_position", 10);

// Subscriber for model states
    ros::Subscriber sub = nh.subscribe<gazebo_msgs::ModelStates>("/gazebo/model_states", 10, UWBcallback);
```

Παράλληλα, μέσα στην βασική συνάρτηση οφείλουν να ορισθούν και τα σενάρια σύμφωνα με τα οποία θα πραγματοποιηθούν οι προσομοιώσεις. Τα σενάρια αυτά έχουν ως στόχο τον προσδιορισμό μιας πορείας την οποία θα ακολουθήσει το drone και είναι πέντε στον αριθμό. Η ποιότητα κατά την παρακολούθηση αυτής της πορείας από τους τοποθετημένους αισθητήρες θα είναι αυτή που θα καθορίσει την αποτελεσματικότητα του κώδικα. Στην συνέχεια, παρατίθεται ο κώδικας ο οποίος δημιουργεί το πρώτο σενάριο με την μορφή παραδείγματος. Αυτό είναι μια απλή απογείωση σε ύψος δύο μέτρων και παραμονή εκεί. Τα υπόλοιπα σενάρια των προσομοιώσεων θα δοθούν με λεπτομέρεια στο επόμενο κεφάλαιο.

```
//Senario1 takeoff
 pose.pose.position.x = 0.0;
pose.pose.position.y = 0.0;
 pose.pose.position.z = 2.0;
```

Έχοντας ολοκληρώσει τα βασικά ορίσματα και τα σενάρια πρέπει να γραφεί ο κώδικας ο οποίος θα συντελεί στην εκκίνηση του drone και θα καθορίζει την απομακρυσμένη λειτουργία του. Αναλυτικότερα, αυτός ,αρχικά, δημοσιεύει συνεχόμενα σημεία αναφοράς θέσης για λόγους ασφάλειας πριν από την εκκίνηση, χρησιμοποιώντας έναν βρόχο που επαναλαμβάνεται εκατό φορές. Στη συνέχεια, δημιουργεί ένα αίτημα για να θέσει τη λειτουργία του drone σε απομακρυσμένη (Off board) και επιπλέον ένα αίτημα για να το οπλίσει. Αυτά παρέχονται αναλυτικά παρακάτω.

```
//send a few setpoints before starting for safety
for(int i = 100; ros::ok() && i > 0; --i){
    local_pos_pub.publish(pose);
    ros::spinOnce();
    rate.sleep();
}
```

```
//This code makes the offboard situation and arms the drone

   mavros_msgs::SetMode offb_set_mode;
   offb_set_mode.request.custom_mode = "OFFBOARD";

   mavros_msgs::CommandBool arm_cmd;
   arm_cmd.request.value = true;

   ros::Time last_request = ros::Time::now();

while(ros::ok()){
   if( current_state.mode != "OFFBOARD" &&
      (ros::Time::now() - last_request > ros::Duration(5.0))){
      if( set_mode_client.call(offb_set_mode) &&
         offb_set_mode.response.mode_sent){
         ROS_INFO("Offboard enabled");
      }
      last_request = ros::Time::now();
   } else {
      if( !current_state.armed &&
         (ros::Time::now() - last_request > ros::Duration(5.0))){
         if( arming_client.call(arm_cmd) &&
            arm_cmd.response.success){
            ROS_INFO("Vehicle armed");
         }
         last_request = ros::Time::now();
      }
   }
```

Ολοκληρώνοντας, στην βασική συνάρτηση παρατίθενται ορισμένες εντολές οι οποίες δημοσιεύουν τα μηνύματα των αισθητήρων αλλά και άλλες οι οποίες πραγματοποιούν εκκίνηση της λειτουργίας των καταγραφών και των εκδοτών. Οι εντολές αυτές παρέχονται στη συνέχεια με την μορφή κώδικα.

```
 // Publish all the imu data
```

```
imu_pub.publish(acc_data);

imu_pub1.publish(vel_data);

imu_pub2.publish(position_data);



 // These commands make the publishers run

ros::spinOnce();

loop_rate.sleep();
```

### 5.2.3 Συμπληρωματικοί κώδικες

Η εκτέλεση του κόμβου δεν έχει τη δυνατότητα να πραγματοποιηθεί από μόνης της. Για να μπορέσει να λειτουργήσει ο κώδικας ο οποίος δημιουργήθηκε με σκοπό την πλοήγηση του drone και την τοποθέτηση αισθητήρων θα πρέπει να γραφούν τρία συμπληρωματικά αρχεία κώδικα. Αυτά καθορίζουν τον ορισμό των βιβλιοθηκών που απαιτούνται για την λειτουργία, την διαδικασία της εκκίνησης ολόκληρου του συστήματος και τα χαρακτηριστικά του drone iris που χρησιμοποιείται.

Ξεκινώντας την αναφορά, το πρώτο αρχείο που αξίζει να συζητηθεί είναι το CMakeList. Αυτό βρίσκεται σε μορφή .txt και ορίζει τη διαδικασία κατασκευής του κόμβου. Πιο αναλυτικά, αυτό φαίνεται να καθορίζει τις εξαρτήσεις του πακέτου από άλλες βιβλιοθήκες ορίζει τις ρυθμίσεις μεταγλώττισης έτσι ώστε να μπορεί να λειτουργεί η γλώσσα C++. Παράλληλα, προσδιορίζει τα μηνύματα που πρέπει να παραχθούν κατά τη διάρκεια της κατασκευής. Ειδικότερα, το αρχείο βρίσκει και περιλαμβάνει τα απαραίτητα πακέτα του ROS, όπως τα sensor_msgs, roscpp και geometry_msgs αλλά και ορίζει το φάκελο που πρέπει να χρησιμοποιηθεί για την εκτέλεση εντολών γραμμικής άλγεβρας. Τέλος, είναι υπεύθυνο για την ενοποίηση όλων των βιβλιοθηκών μεταξύ τους και την ορθή λειτουργία του κόμβου.

Συνεχίζοντας, σημαντικό αρχείο για την εκτέλεση του κώδικα είναι το αρχείο εκκίνησης (launch file) γραμμένο σε γλώσσα XML. Αυτό χρησιμοποιείται για την έναρξη και τη διαμόρφωση του περιβάλλοντος προσομοίωσης και του ελέγχου του drone. Αναλυτικότερα, το αρχείο ενσωματώνει το αρχείο εκκίνησης της εταιρείας Px4, το οποίο θέτει σε λειτουργία τον κόμβο MAVROS για επικοινωνία της εφαρμογής του Gazebo με το λογισμικό πτήσης Επιπλέον, ορίζει την εκκίνηση του κόμβου καθώς και είναι υπεύθυνο για τον καθορισμό παραμέτρων που χρησιμοποιούνται για την ακριβή χρονομέτρηση στην προσομοίωση. Με αυτόν τον τρόπο, το αρχείο εκκίνησης διασφαλίζει τη σωστή διαμόρφωση και έναρξη του περιβάλλοντος προσομοίωσης και ελέγχου του drone.

Ολοκληρώνοντας την αναφορά στα συμπληρωματικά αρχεία είναι χρήσιμη η περιγραφή αυτού με όνομα iris.sdf. Το αρχείο αυτό είναι γραμμένο σε γλώσσα προγραμματισμού XML και σκοπός του είναι να προσδιορίζει ακριβώς το μη επανδρωμένο εναέριο όχημα. Ειδικότερα, αυτό διαθέτει πληροφορίες οι οποίες διαμορφώνουν το βάρος και τα γεωμετρικά χαρακτηριστικά του drone. Παράλληλα, μέσω αυτού δίνονται όλες οι απαραίτητες λεπτομέρειες οι οποίες καθορίζουν τις αρθρώσεις του αλλά και των αισθητήρων που είναι ενσωματωμένοι σε αυτό. Αυτά εισάγονται στο αρχείο με την μορφή πρόσθετου τυποποιημένου κώδικα και είναι υπεύθυνα για ολόκληρη την λειτουργία του συστήματος. Αυτό γίνεται καθώς, η προσαρμογή αυτή των αισθητήρων παράγει τα μηνύματα για την αξιοποίηση των καταγραφών και εκδοτών που έχουν συμπεριληφθεί στον κόμβο.

# 6 ΟΙ ΠΡΟΣΟΜΟΙΩΣΕΙΣ

Σκοπός του παρόντος κεφαλαίου αποτελεί η κατανόηση της λειτουργικότητας του κώδικα μέσω της διαδικασίας των προσομοιώσεων. Πιο αναλυτικά, έχουν κατασκευασθεί πέντε υποθετικά σενάρια πτήσης τα οποία εξετάζουν την ικανότητα των αισθητήρων που έχουν μοντελοποιηθεί να ακολουθούν τις εκάστοτε κινήσεις του εναέριου οχήματος. Οι αισθητήρες οι οποίοι θεωρούνται χρήσιμοι για την καλύτερη αντίληψη της κατάστασης του ρομποτικού συστήματος είναι το IMU και το UWB,ανάλυση της λειτουργίας των οποίων έχει πραγματοποιηθεί σε προηγούμενα κεφάλαια. Ταυτόχρονα, οι κινήσεις που εκτελούνται κατά την υλοποίηση των σεναρίων προσομοίωσης προσπαθούν να λάβουν υπόψη όλα τα ενδεχόμενα μετατοπίσεων ενός drone σε εσωτερικό βιομηχανικό χώρο.

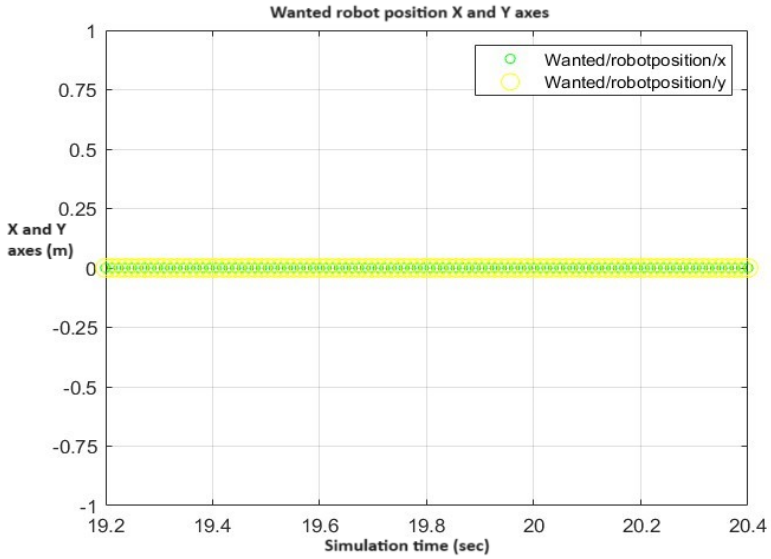
Με στόχο την καλύτερη κατανόηση, παρέχονται σε κάθε σενάριο διαγράμματα τα οποία καταδεικνύουν αφενός την επιθυμητή κατάσταση και αφετέρου την κατάσταση την οποία μπορεί να αντιληφθεί ο αισθητήρας. Τα διαγράμματα αυτά απευθύνονται σε συγκεκριμένα στιγμιότυπα της κίνησης και δεν την καταγράφουν ολόκληρη. Αυτό συμβαίνει καθώς το πρόγραμμα που χρησιμοποιείται για τον σχεδιασμό τους έχει την ικανότητα να καταγράφει την κίνηση κατά την διάρκεια της προσομοίωσης και όχι να εμφανίζει ένα τελικό διάγραμμα σε συνολικό χρόνο. Την ίδια στιγμή, θεωρείται πρέπον να αποδοθούν και ενοποιημένα διαγράμματα των παραπάνω περιπτώσεων έτσι ώστε να υπάρχει όσο τον δυνατόν μεγαλύτερη πληρότητα.

## 6.1 Σενάριο 1 : Απλή απογείωση και παραμονή στο ίδιο σημείο και ύψος

Πραγματοποιώντας αναφορά στο πρώτο σενάριο, αυτό αποτελεί μια απλή απογείωση του drone και παραμονή του στο ίδιο ύψος. Πιο αναλυτικά, δίνεται στο εναέριο όχημα η εντολή να μεταβεί από την αρχική του θέση η οποία έχει ορισθεί στο σημείο (0,0,0) στην θέση (0,0,2) χωρίς να εκτελέσει άλλη κίνηση. Συνεπώς, εκτελείται μια απλή κατακόρυφη απογείωση μέχρι το ύψος των 2m

### 6.1.1 Διαγράμματα για τον αισθητήρα UWB για το σενάριο 1

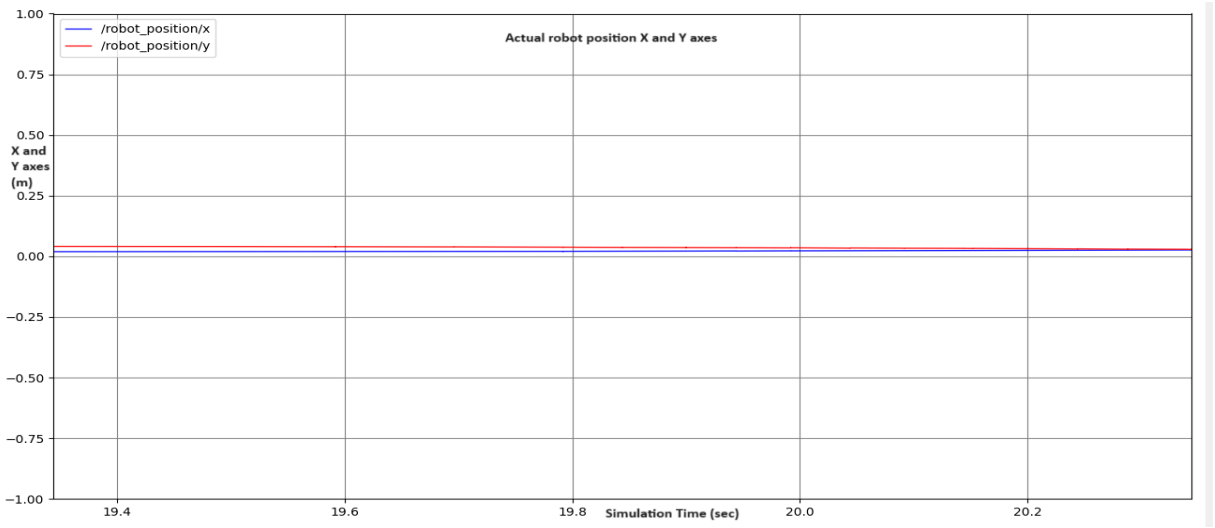
Κατά την εκτέλεση αυτού του σεναρίου, το drone παραμένει στην ίδια ακριβώς θέση κατά τους άξονες x και y. Επομένως, ο αισθητήρας UWB θα πρέπει έχει ενδείξεις οι οποίες θα παραμένουν σταθερά στην μηδενική κατάσταση. Αυτό σχηματικά απεικονίζεται στην Εικόνα 6.1-1.



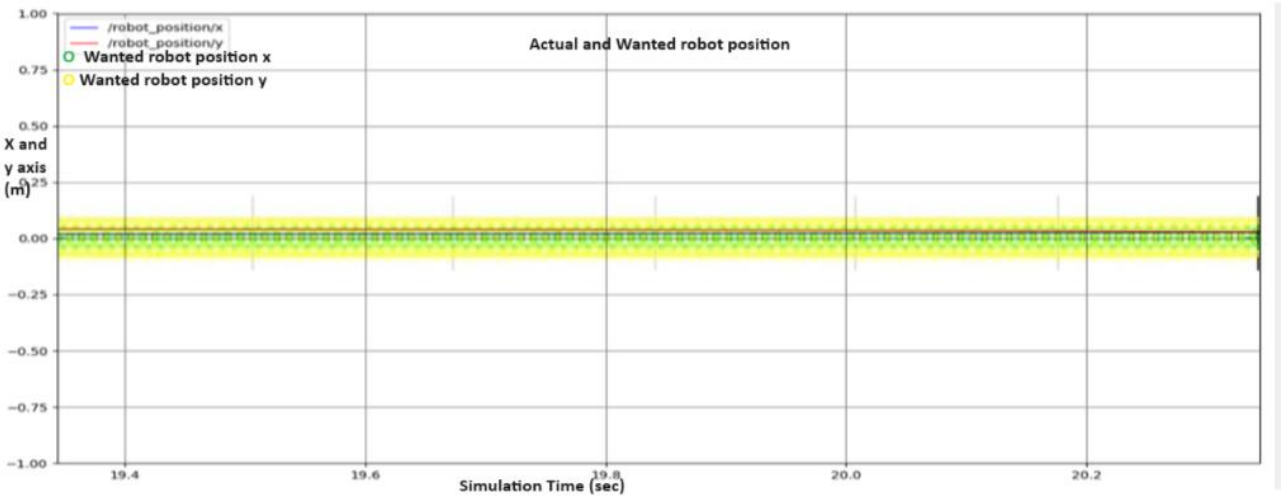

**Εικόνα 6.1-1: Επιθυμητή μέτρηση αισθητήρα UWB στο σενάριο 1**

Εκτελώντας τον κώδικα ο οποίος είναι δομημένος με στόχο να προσομοιώνει τον αισθητήρα UWB λαμβάνεται το αποτέλεσμα που φαίνεται στην Εικόνα 6.1-2. Ο αισθητήρας που χρησιμοποιείται για χάρη της προσομοίωσης εντοπίζει με αρκετή ακρίβεια την θέση του drone καθώς αυτή αποκλίνει ελάχιστα από την

μηδενική. Επιπλέον, η απόκλιση αυτή μπορεί να θεωρηθεί φυσιολογική καθώς δύναται να υπάρχουν μικρές μετακινήσεις του εναέριου οχήματος κατά την παραμονή του σε σταθερή θέση. Στην Εικόνα 6.1-3 φαίνεται η ενοποίηση των διαγραμμάτων που παρουσιάζουν την επιθυμητή και την τελική θέση του μη επανδρωμένου εναέριου οχήματος.

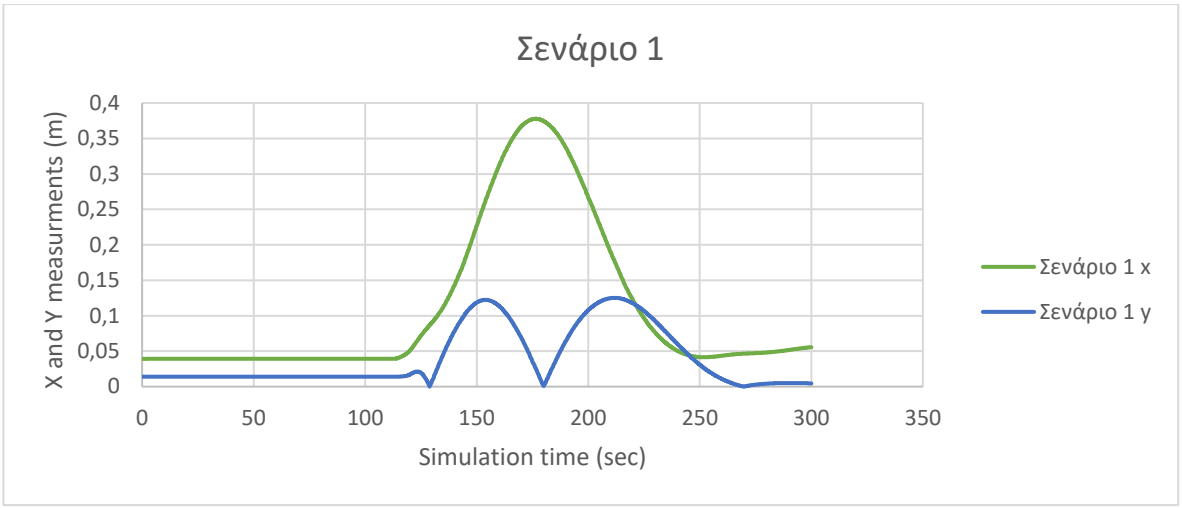



**Εικόνα 6.1-2:Μέτρηση αισθητήρα UWB στο σενάριο 1**




**Εικόνα 6.1-3: Ενοποίηση επιθυμητής και πραγματικής κατάστασης αισθητήρα UWB για σενάριο 1**

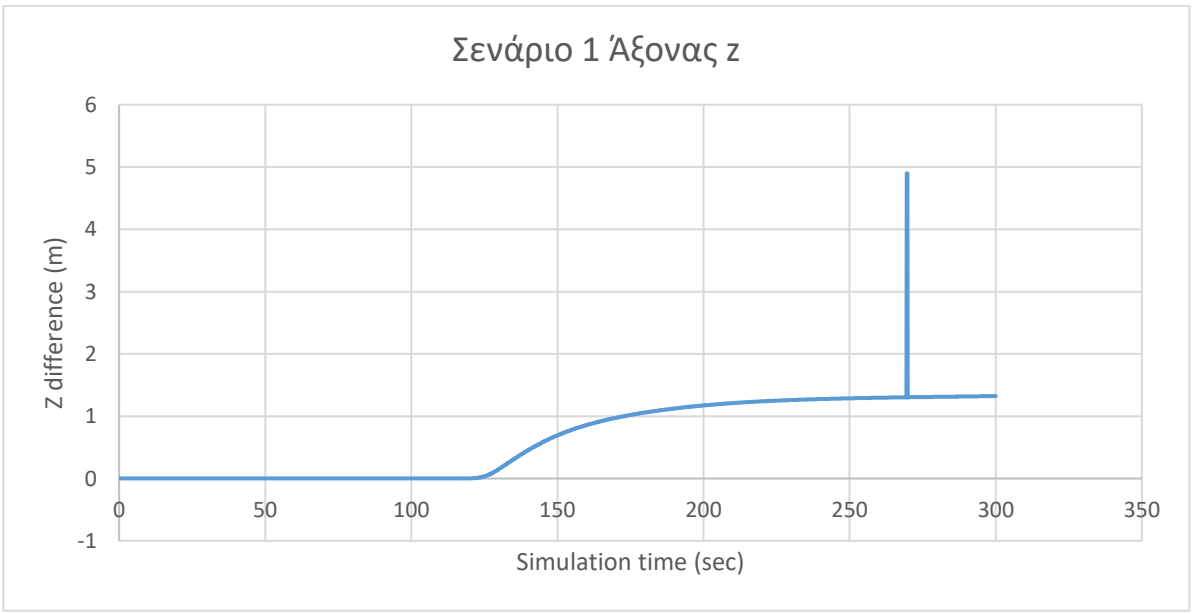
Εξαιτίας του γεγονότος ότι οι παραπάνω εικόνες παρουσιάζουν στιγμιότυπα της κίνησης κρίθηκε σημαντικό να δημιουργεί και ένα διάγραμμα το οποίο θα απεικονίζει την συνολική κίνηση του drone κατά την εκτέλεση αυτού του σεναρίου. Το διάγραμμα αυτό φαίνεται στην Εικόνα 6.1-4.




**Εικόνα 6.1-4: Συνολική κίνηση του drone στο σενάριο 1**

Με χρήση του παραπάνω διαγράμματος φαίνεται η ακριβής παρακολούθηση του αισθητήρα κατά την διεξαγωγή του πρώτου σεναρίου. Ειδικότερα, αυτός αντιλαμβάνεται την θέση του εναέριου οχήματος όπως αυτή έχει καθοριστεί. Η μεγαλύτερη απόκλιση η οποία παρουσιάζει από την επιθυμητή θέση είναι αυτή των 0,4 m στις μετρήσεις του άξονα x'x αλλά δείχνει να την ξεπερνά γρήγορα και να επανέρχεται στις βασικές του τιμές. Συνεπώς, συμπεραίνεται ότι ο κώδικας που μοντελοποιεί τον αισθητήρα UWB μπορεί να παρακολουθεί τις θέσεις του drone στους άξονες x και y με σημαντική ακρίβεια.

Αδυναμία του κώδικα, ωστόσο, εμφανίζεται στην μέτρηση του άξονα z'z, δηλαδή του ύψους στο οποίο βρίσκεται το μη επανδρωμένο εναέριο όχημα. Ο υπολογισμός αυτού του παράγοντα δεν είναι καθόλου ακριβής και απέχει αρκετά από την εντολή που δίνεται στο σύστημα. Επομένως, επιλέγεται να παρουσιαστεί η διαφορά των επιμέρους υψών από το αρχικό. Αυτό απεικονίζεται στην Εικόνα 6.1-5.



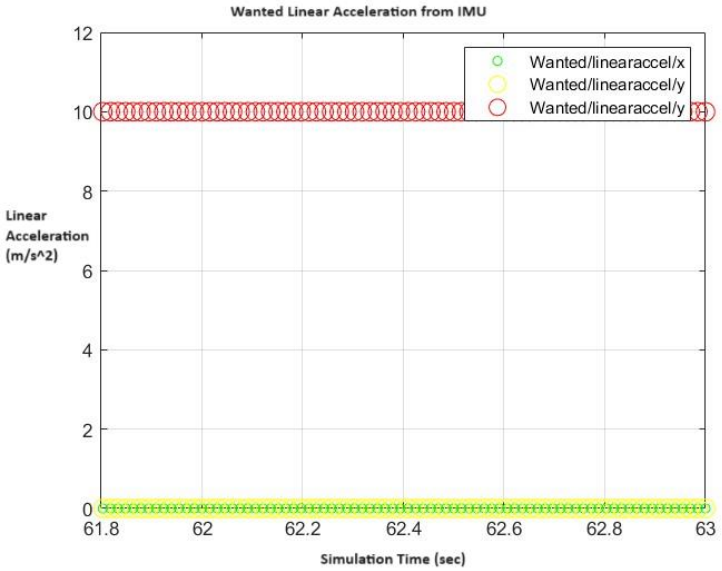

**Εικόνα 6.1-5: Διαφορά των μετρήσεων του άξονα z στο σενάριο 1**

Από το παραπάνω διάγραμμα προκύπτει ότι ο κώδικας που γράφηκε για τον αισθητήρα UWB δεν μπορεί να χρησιμοποιηθεί για να υπολογίζει το ύψος του οχήματος. Αυτό συμβαίνει καθώς δεν προσεγγίζει σε καμία περίπτωση την τιμή των 2 m η οποία έχει οριστεί για το σύστημα. Ωστόσο, έχει την ικανότητα να αντιλαμβάνεται την αλλαγή της κινητικής κατάστασης του drone και να την καταγράφει ικανοποιητικά.

### 6.1.2 Διαγράμματα για τον αισθητήρα IMU για το σενάριο 1

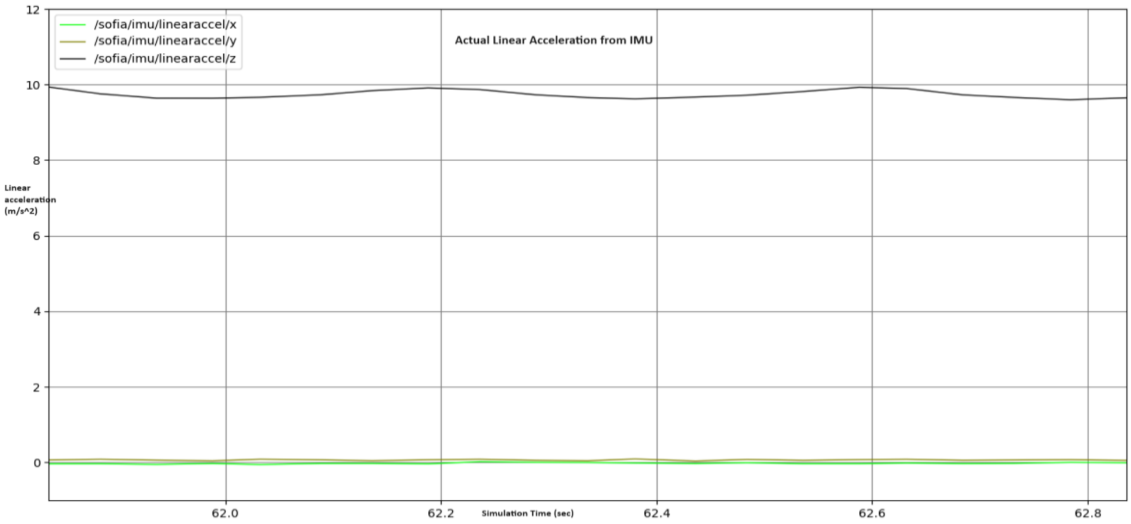
Σημαντικός για την πλήρη επίγνωση της κατάστασης του οχήματος είναι ο αισθητήρας IMU. Αυτός έχει την ικανότητα να μετρά τόσο την γραμμική επιτάχυνση όσο και την ταχύτητα του drone κάθε χρονική στιγμή.

Ξεκινώντας την αναφορά από το μέγεθος της γραμμικής επιτάχυνσης αυτή λαμβάνει καθορισμένες τιμές. Πιο συγκεκριμένα, εξαιτίας της κίνησης του drone στο άξονα z καθώς και της τυποποιημένης μετακίνησής του από το πρόγραμμα αναμένεται να υπάρχει γραμμική επιτάχυνση κατά τον άξονα της πτήσης ίση με 10 $m/s^2$ ενώ αυτές που αναφέρονται στους άλλους δύο άξονες να παραμένουν στο μηδέν. Επομένως, το διάγραμμα με τις αναμενόμενες τιμές παρουσιάζεται στην Εικόνα 6.1-6.

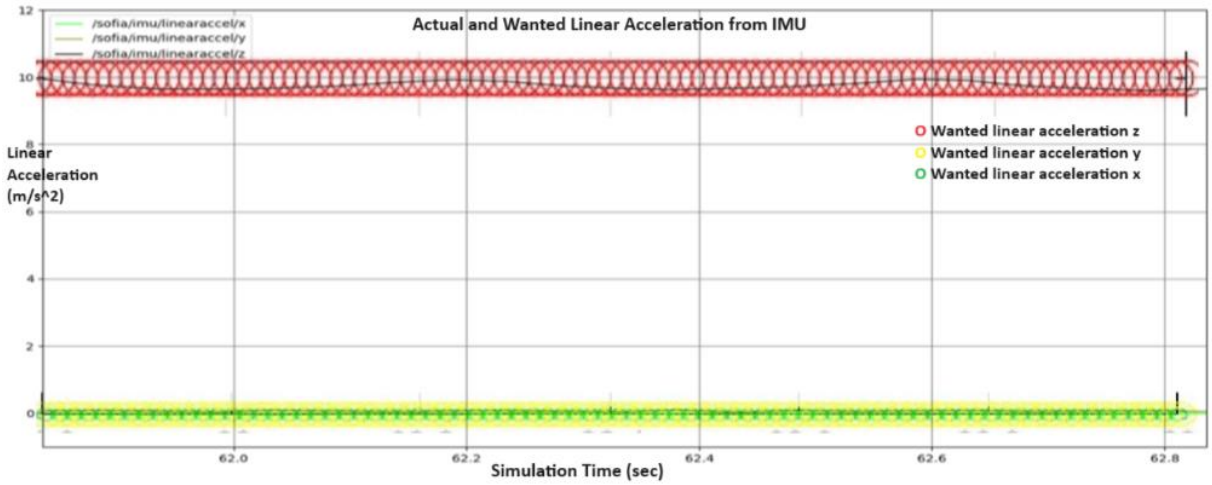



**Εικόνα 6.1-6: Διάγραμμα επιθυμητών τιμών γραμμικής επιτάχυνσης για σενάριο 1**

Εκτελώντας το αρχείο του κώδικα και λαμβάνοντας τα δημοσιευμένα αποτελέσματα λαμβάνεται το διάγραμμα το οποίο φαίνεται στην Εικόνα 6.1-7 για μια τυχαία χρονική στιγμή. Παράλληλα, στην Εικόνα 6.1-8 παρουσιάζεται και η ενοποίηση των δύο διαγραμμάτων με στόχο την απευθείας σύγκριση.




**Εικόνα 6.1-7: Γραμμική επιτάχυνση προσομοίωσής για σενάριο 1**



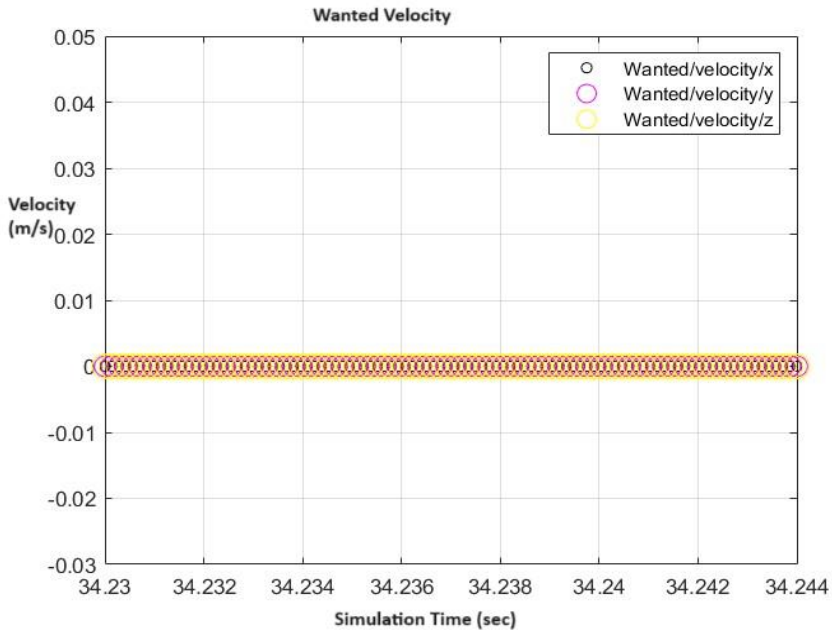

**Εικόνα 6.1-8: Ενοποιημένο διάγραμμα επιθυμητής και μετρούμενης επιτάχυνσης για σενάριο1**

Ο αισθητήρας IMU έχει την ικανότητα να αντιλαμβάνεται σε μεγάλο βαθμό την κατάσταση του συστήματος. Αναλυτικότερα, κατά την διεξαγωγή των μετρήσεων παρατηρείται μια διακύμανση στο μέγεθος της γραμμικής επιτάχυνσης του άξονα z. Αυτή φαίνεται να έχει περιοδική μορφή γεγονός που σημαίνει ότι πρόκειται για ένα συστηματικό σφάλμα του προσομοιωμένου αισθητήρα. Με στόχο την κατανόηση του μεγέθους του σφάλματος αξιοποιείται η σχέση:

$$Mp\% = \frac{z_{\tau\varepsilon\lambda} - z_{\alpha\rho\chi}}{z_{\alpha\rho\chi}} \cdot 100\% \qquad (6.1)$$

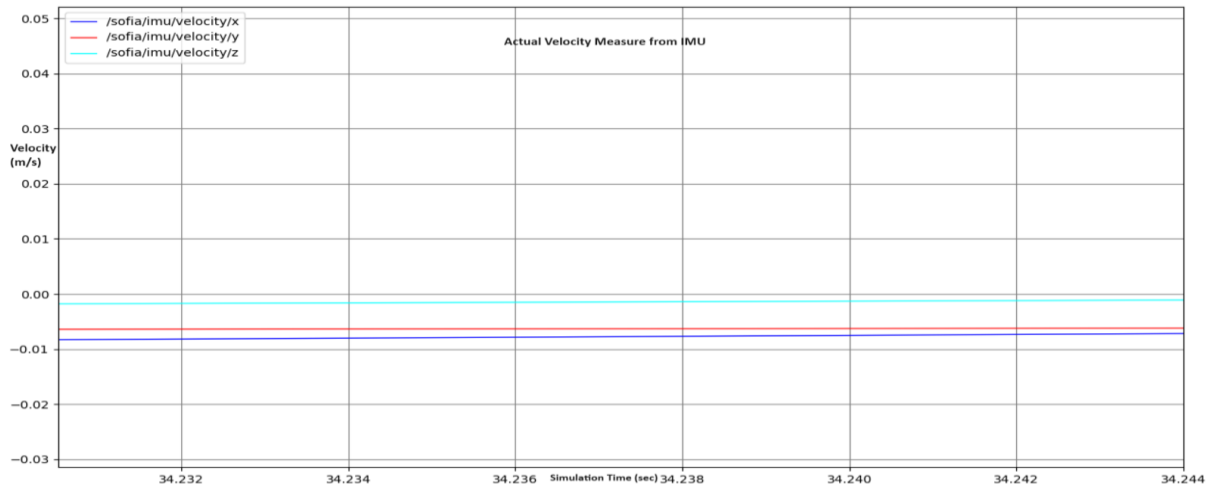
Αντικαθιστώντας τα μεγέθη τα οποία φαίνονται στην Εικόνα 6.1-7 το σφάλμα υπολογίζεται περίπου ίσο με 2%, τιμή που κρίνεται ως λογική και επιτρεπτή για το σύστημα.

Έχοντας ολοκληρώσει την μέτρηση της γραμμικής επιτάχυνσης θεωρείται χρήσιμο να επισημανθεί και το μέγεθος της ταχύτητας του drone. Αυτό μετριέται και πάλι στους τρείς άξονες x, y,z από τον ίδιο αισθητήρα IMU. Από την στιγμή που το εναέριο όχημα δεν μετακινείται αλλά διατηρεί σταθερή την θέση του σε όλους τους άξονες αναμένεται ταχύτητα ίση με το μηδέν. Με αυτόν τον τρόπο, το διάγραμμα το οποίο μπορεί να περιγράψει την επιθυμητή κατάσταση κατά την διάρκεια της εκκίνησης του οχήματος φαίνεται στην Εικόνα 6.1-9. Ωστόσο, την ίδια στιγμή σημειώνεται ως εντελώς λογικό να εμφανίζονται στιγμιαίες επιταχύνσεις και επιβραδύνσεις στο διάγραμμα. Αυτό γίνεται καθώς το drone που μελετάται θα πρέπει να αναπτύξει ταχύτητα και έπειτα να την ελαττώσει έτσι ώστε να φτάσει την τελική θέση που του επιβάλλει η κίνησή του.

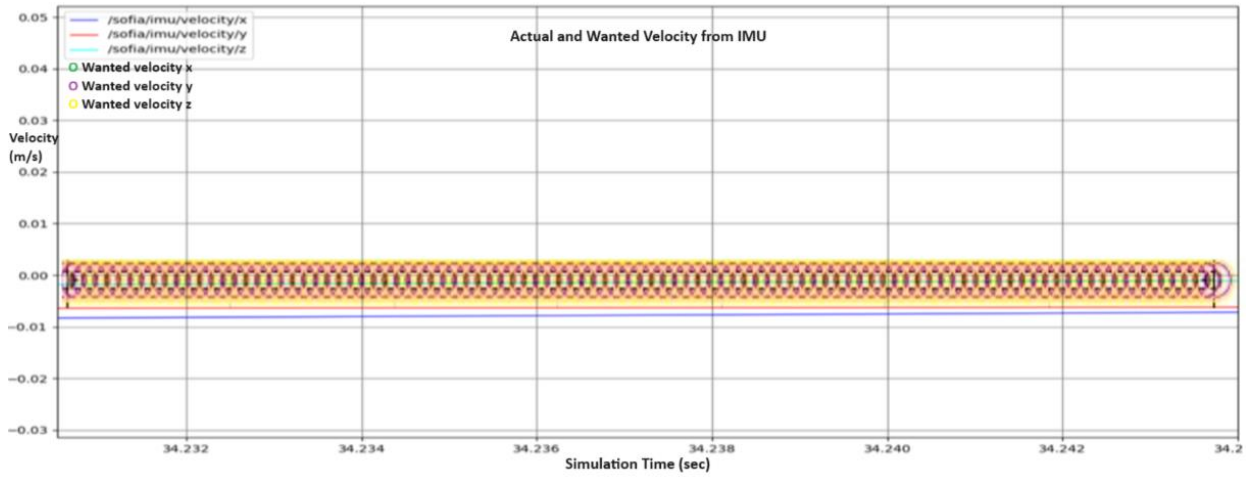



**Εικόνα 6.1-9: Επιθυμητή γραμμική ταχύτητα οχήματος για σενάριο 1**

Έχοντας αποδώσει σχηματικά την επιθυμητή κατάσταση κατά την διάρκεια της εκκίνησης πρέπει να αποδοθεί και αυτή η οποία υπολογίζεται κατά την προσομοίωση. Αυτή παρουσιάζεται στην Εικόνα 6.1-10. Παρατηρώντας το διάγραμμα, είναι φανερό ότι υπάρχουν μικρές αποκλίσεις από την τιμή μηδέν. Αυτές φαίνεται να είναι λιγότερες από 0,01 m/s και δύναται να οφείλονται στο φαινόμενο των στιγμιαίων επιταχύνσεων και επιβραδύνσεων που αναφέρθηκε. Παράλληλα, τα παραπάνω μπορούν να γίνουν αντιληπτά με βάση και το ενοποιημένο διάγραμμα που φαίνεται στην Εικόνα 6.1-11

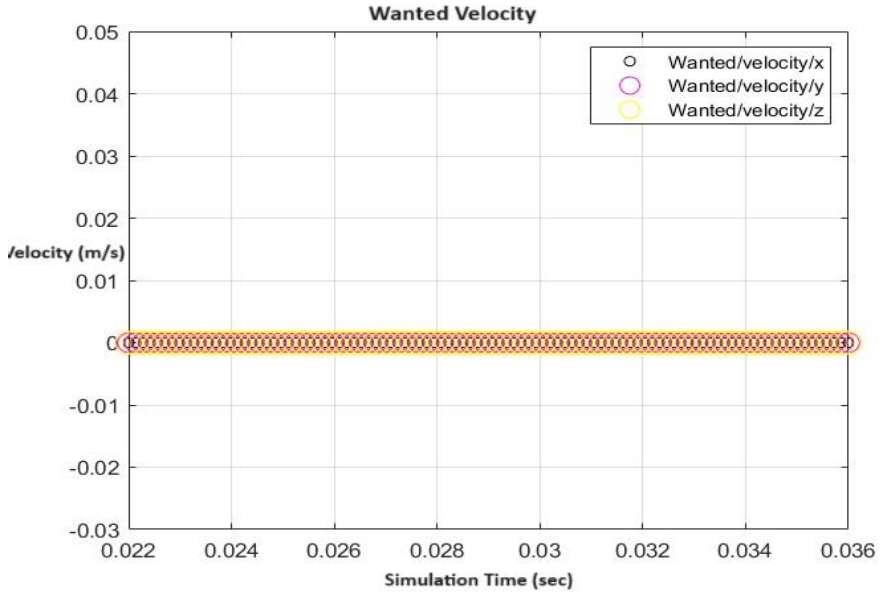



**Εικόνα 6.1-10: Γραμμική ταχύτητα κατά την προσομοίωση στο σενάριο 1**



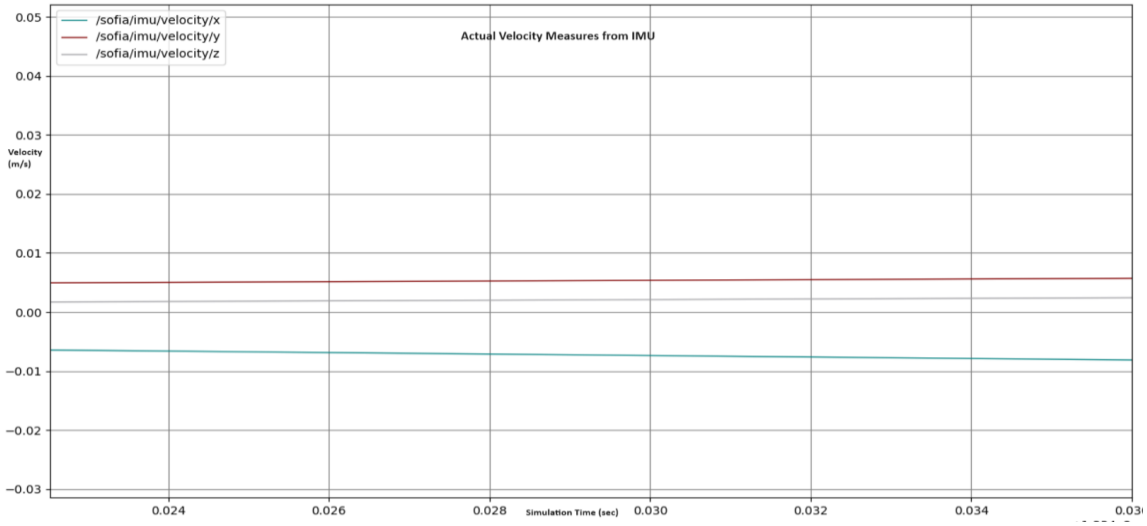

**Εικόνα 6.1-11: Ενοποιημένο διάγραμμα γραμμικής ταχύτητας κατά την εκκίνηση για σενάριο 1**

Με στόχο την κατανόηση της μέτρησης της ταχύτητας και την αντίληψη της ορθής λειτουργίας του αισθητήρα θεωρείται πρέπον να ληφθεί άλλο ένα στιγμιότυπο. Αυτό λαμβάνεται έπειτα από παρέλευση αρκετού χρόνου προσομοίωσης με σκοπό να εξετασθεί η ακρίβεια των μετρήσεων όταν η θέση του drone θα είναι έχει παραμείνει σταθερή. Επομένως, η επιθυμητή μέτρηση η οποία πρέπει να ληφθεί παρουσιάζεται στην Εικόνα 6.1-12.

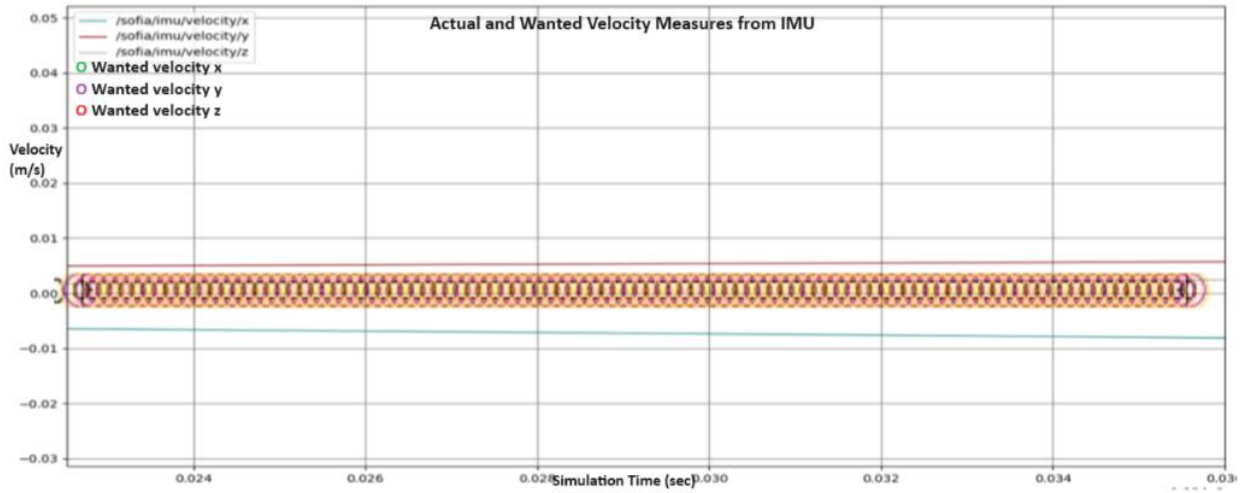



**Εικόνα 6.1-12: Διάγραμμα επιθυμητής ταχύτητας σεναρίου 1 μετά από παρέλευση χρόνου**

Επιπλέον, η μέτρηση που λαμβάνεται κατά την διάρκεια της προσομοίωσης αλλά και η ενοποίηση των διαγραμμάτων φαίνονται στην Εικόνα 6.1-13 και στην Εικόνα 6.1-14 αντίστοιχα. Παρατηρώντας τις εικόνες γίνεται αντιληπτό ότι παρά την σταθερότητα της θέσης οι μεταβολές στις τιμές της ταχύτητας συνεχίζουν να λαμβάνουν χώρα. Ωστόσο, αυτές είναι και πάλι λιγότερες από 0,01 m/s και δεν στερούν πολλά από την ακρίβεια του αισθητήρα.




**Εικόνα 6.1-13: Προσομοιωμένη ταχύτητα σεναρίου 1 μετά από παρέλευση χρόνου**



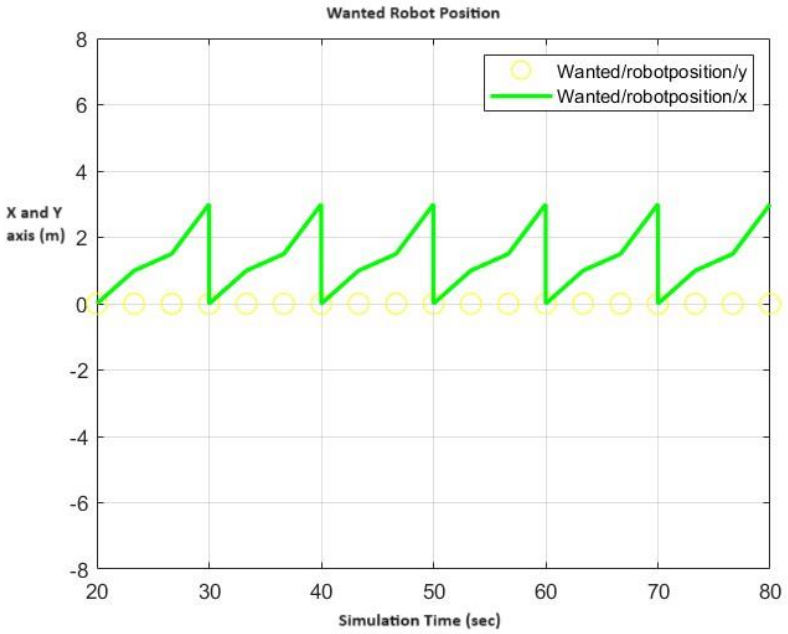

**Εικόνα 6.1-14: Ενοποιημένο διάγραμμα ταχύτητας με παρέλευση χρόνου για σενάριο 1**

## 6.2 Σενάριο 2: Ευθύγραμμη τροχιά με μετακίνηση κατά τον άξονα x'x.

Η κίνηση που εκτελείται κατά την ανάπτυξη του δεύτερου σεναρίου περιλαμβάνει τέσσερα σημεία το οποία δημιουργούν μια ευθύγραμμη τροχιά. Πιο συγκεκριμένα, το drone ξεκινά από το σημείο (0,0,2) και συνεχίζει στο σημείο (1,0,2). Έπειτα, αυτό πηγαίνει στο σημείο (1.5,0,2) και τελικά καταλήγει στη θέση (3,0,2). Η κίνηση αυτή του drone είναι περιοδική.

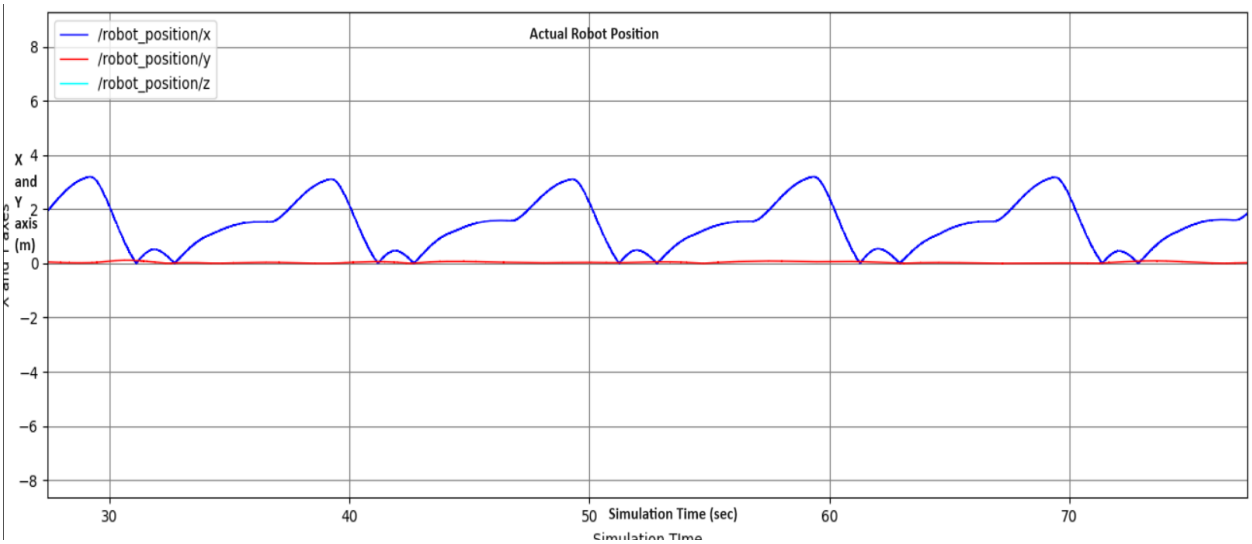
### 6.2.1 Ο αισθητήρας UWB κατά το σενάριο 2

Ξεκινώντας την μελέτη του δεύτερου σεναρίου ελέγχεται η λειτουργία του κώδικα ο οποίος περιγράφει τον αισθητήρα UWB. Η τροχιά που ζητείται να ακολουθηθεί είναι ευθύγραμμη και περιοδική και φαίνεται στην Εικόνα 6.2-1.

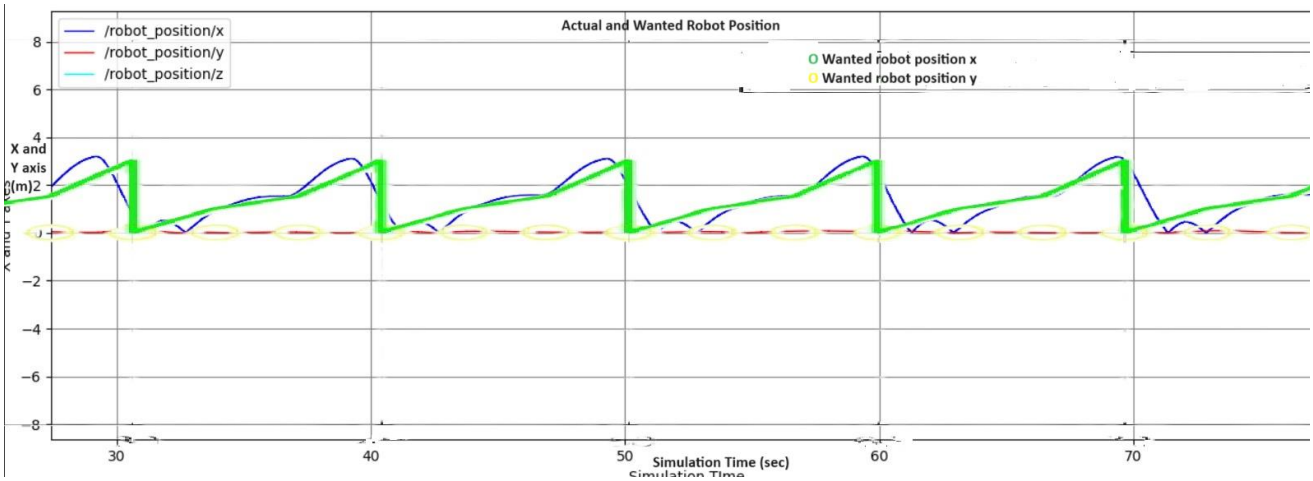



**Εικόνα 6.2-1: Επιθυμητή τροχιά σεναρίου 2**

Την ίδια στιγμή, η τροχιά η οποία προκύπτει από τον προσομοιωμένο αισθητήρα παρουσιάζεται στην Εικόνα 6.2-2. Παράλληλα, για καλύτερη κατανόηση της απόκλισης παρατίθεται η Εικόνα 6.2-3.



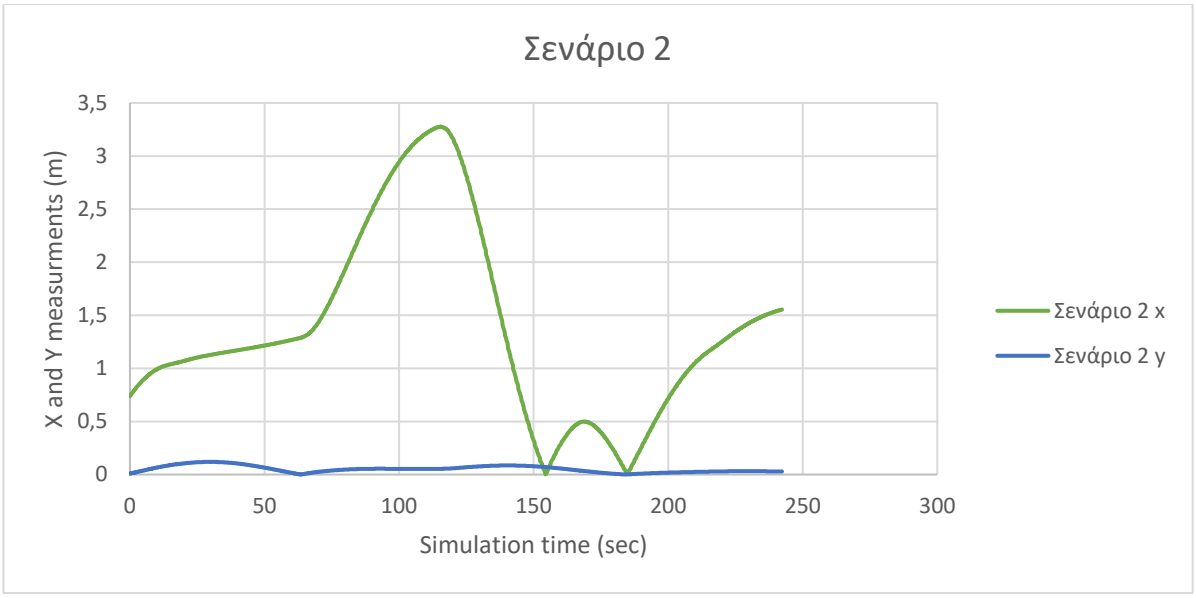

**Εικόνα 6.2-2: Τροχιά από προσομοιωμένο αισθητήρα UWB για σενάριο 2**




**Εικόνα 6.2-3: Επιθυμητή και προσομοιωμένη τροχιά για σενάριο 2**

Ο προσομοιωμένος αισθητήρας UWB αντιλαμβάνεται σε μεγάλο βαθμό την τροχιά την οποία εκτελεί το drone. Βασική διαφορά των δύο καταστάσεων αποτελούν οι μεταβάσεις. Πιο αναλυτικά, η επιθυμητή τροχιά έχει πιο απότομες μεταβιβάσεις από την μία θέση στην άλλη γεγονός που γίνεται κατανοητό από την κλίση των διαγραμμάτων. Την ίδια στιγμή, στην προσομοίωση φαίνεται να υπάρχει δύο φορές επιστροφή στην μηδενική θέση φαινόμενο που δεν ορίζεται κατά τον καθορισμό της τροχιάς.

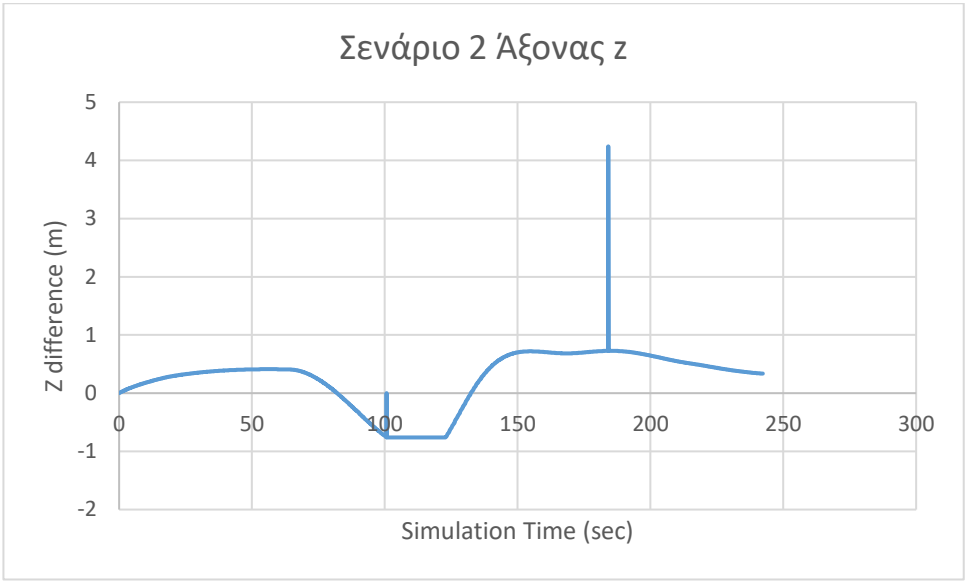
Με στόχο την δημιουργία μιας πιο ολοκληρωμένης αντίληψης για τις μετρήσεις του αισθητήρα UWB παρέχεται ένα διάγραμμα το οποίο απεικονίζει τις μετρήσεις του κατά την εκτέλεση της συνολικής κίνησης του drone. Αυτό φαίνεται στην Εικόνα 6.2-4.




**Εικόνα 6.2-4: Συνολική τροχιά του drone στο σενάριο 2**

Όπως και στα στιγμιότυπα, έτσι και στη συνολική εικόνα παρατηρείται ότι ο αισθητήρας παρακολουθεί με μεγάλη ακρίβεια τις κινήσεις του ρομποτικού συστήματος. Πιο αναλυτικά, όσον αφορά τον άξονα στον οποίο υλοποιούνται οι μετακινήσεις, δηλαδή τον x'x το σύστημα δεν εμφανίζει διακυμάνσεις. Παράλληλα, αυτό δεν απομακρύνεται σημαντικά από τις καθορισμένες τροχιές που διαμορφώνουν την τροχιά. Την ίδια στιγμή, σοβαρές μεταβολές και υπερακοντίσεις δεν εμφανίζονται ούτε στις μετρήσεις του άξονα y'y ο οποίος είναι στατικός για αυτό το σενάριο. Επομένως, η ακρίβεια των μετρήσεων σε μία ευθύγραμμη κίνηση είναι υψηλή.

Επιπλέον, χρήσιμο είναι να καταδειχθούν οι μετρήσεις του ύψους δηλαδή του άξονα z’z. Αυτές αποτελούν αδυναμία του κώδικα καθώς απέχουν πολύ από την καθορισμένη από τον χειριστή τιμή των 2m. Για αυτό το λόγο παρέχονται οι διαφορές με την αρχική θέση. Αυτά φαίνονται στην Εικόνα 6.2-5.

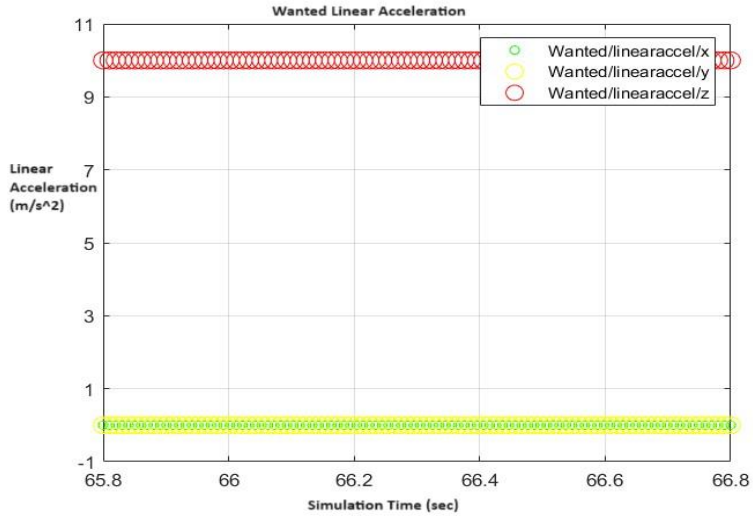



**Εικόνα 6.2-5: Διαφορά υψών στο σενάριο 2**

Οι υπολογισμοί που σημειώνονται με στόχο την μέτρηση του ύψους του drone δεν είναι ικανοποιητικοί. Αυτό έχει ως αποτέλεσμα το ύψος να αποκλίνει σημαντικά από το αναμενόμενο και οι τιμές που παρέχονται δεν μπορούν να χρησιμοποιηθούν. Ωστόσο, με τον υπολογισμό των διαφορών παρατηρείται ότι το ύψος παραμένει αμετάβλητο απλώς σε μία μη αναμενόμενη τιμή. Εξαίρεση σε αυτό αποτελεί η χρονική στιγμή 184,1 sec κατά την οποία η διαφορά αυξάνεται κατακόρυφα και μειώνεται ξανά ακαριαία διατηρώντας φυσιολογικές τιμές.

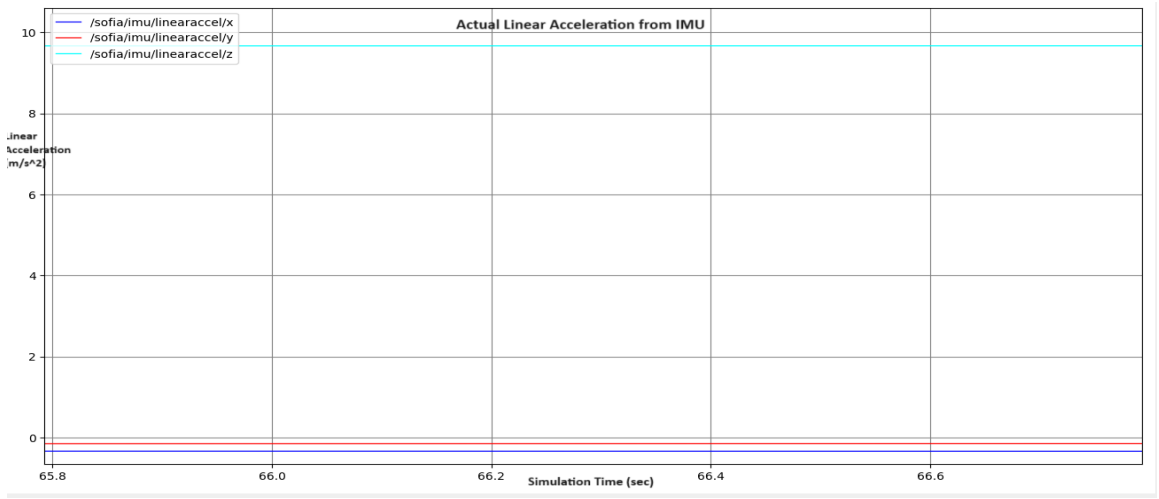
### 6.2.2 Η χρήση του αισθητήρα IMU κατά το σενάριο 2

Κατά την εκτέλεση του παρόντος σεναρίου το drone οφείλει να επιταχύνει και να επιβραδύνει έτσι ώστε να μετακινείται και να φτάνει κάθε φορά στην κατάλληλη θέση. Ωστόσο, όταν αυτό παραμένει ακίνητο θα πρέπει να διατηρεί τις τυποποιημένες τιμές επιτάχυνσης του αισθητήρα IMU που παρέχει η εφαρμογή του Gazebo. Εξαιτίας του γεγονότος ότι τα διαγράμματα απεικονίζουν στιγμιότυπα της κίνησης η επιθυμητή επιτάχυνση για μία τυχαία χρονική στιγμή κατά την διάρκεια της εκκίνησης απεικονίζεται στην Εικόνα 6.2-6.

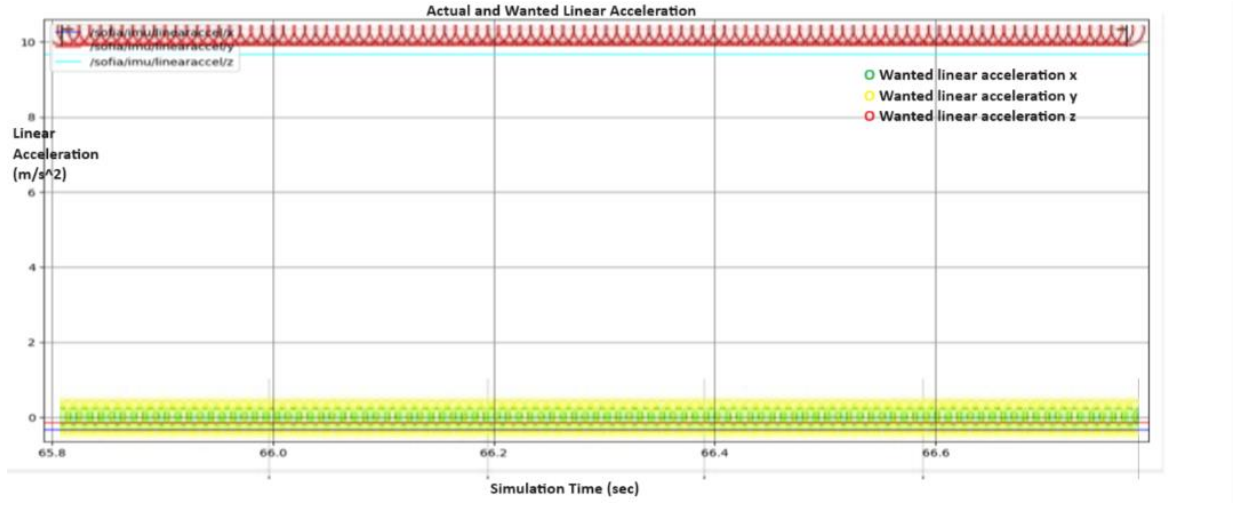



**Εικόνα 6.2-6: Επιθυμητή κατάσταση επιτάχυνσης αισθητήρα IMU για σενάριο 2**

Ταυτόχρονα παρατίθενται τόσο οι επιταχύνσεις που προκύπτουν από την προσομοίωση στην Εικόνα 6.2-7 όσο και το ενοποιημένο διάγραμμα στην Εικόνα 6.2-8. Αυτό δίνει την δυνατότητα άμεσης σύγκρισης και κατανόησης όλων των δυνατών αποκλίσεων.



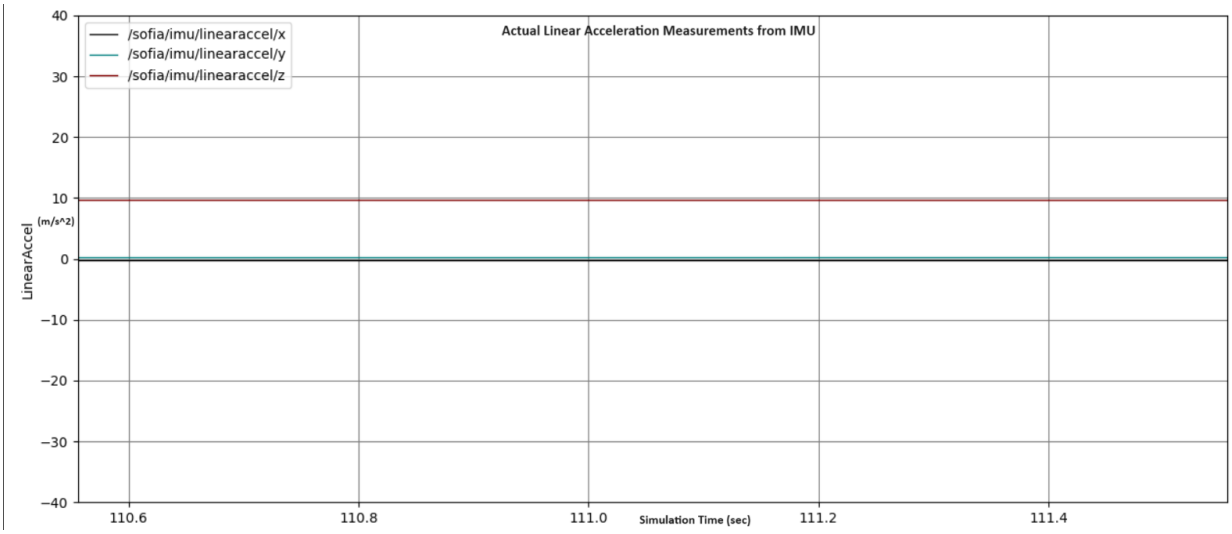

**Εικόνα 6.2-7: Οι επιταχύνσεις κατά την προσομοίωση του σεναρίου 2**



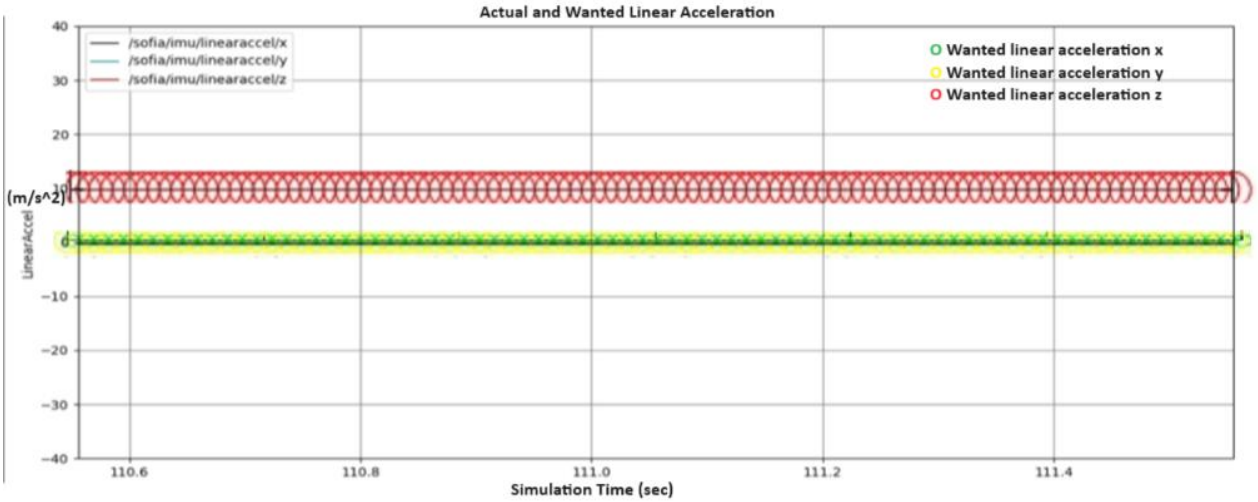

**Εικόνα 6.2-8: Ενοποιημένο διάγραμμα γραμμικής επιτάχυνσης σεναρίου 2**

Παρατηρείται ότι κατά τις χρονικές στιγμές που το drone βρίσκεται στην αρχή της κίνησής του οι τιμές της γραμμικής του επιτάχυνσης δεν βρίσκονται ακριβώς στα σημεία 10 $m/s^2$ και 0 $m/s^2$ τα οποία υποδεικνύει η τυποποίηση. Παρόλα αυτά, οι αποκλίσεις δεν κρίνονται σημαντικές και σχολιάζονται ως λογικές καθώς το σύστημα θα πρέπει να ετοιμαστεί ώστε να εκτελέσει τις απαιτούμενες κινήσεις.

Με στόχο τον έλεγχο της διατήρησης της αστάθειας της γραμμικής επιτάχυνσης του συστήματος εξετάζεται ένα, επιπλέον, στιγμιότυπο κίνησης έπειτα από την παρέλευση ενός αρκετού χρονικού διαστήματος. Η επιθυμητή κατάσταση είναι ίδια με αυτή που απεικονίζεται στην Εικόνα 6.2-6 για κάθε χρονική στιγμή. Η νέα προσομοιωμένη κατάσταση φαίνεται στην Εικόνα 6.2-9. Παράλληλα, για καλύτερη αντιστοιχία της απόκλισης τα δύο διαγράμματα ενοποιούνται και το αποτέλεσμα αυτής της διεργασίας παρουσιάζεται στην Εικόνα 6.2-10.



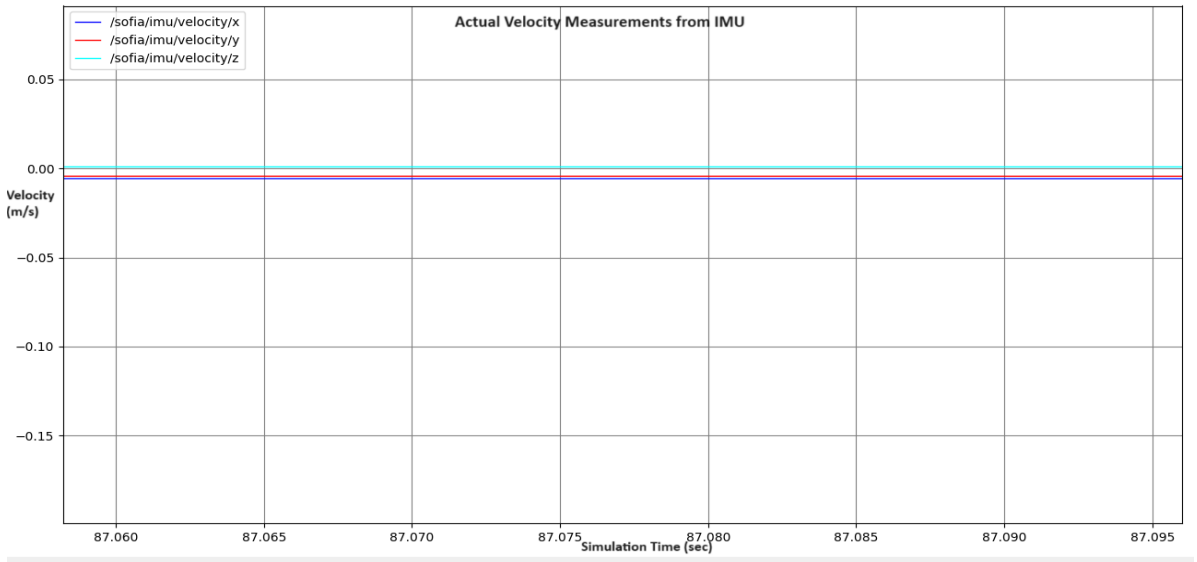

**Εικόνα 6.2-9: Προσομοιωμένη γραμμική επιτάχυνση έπειτα από παρέλευση χρόνου για σενάριο 2**



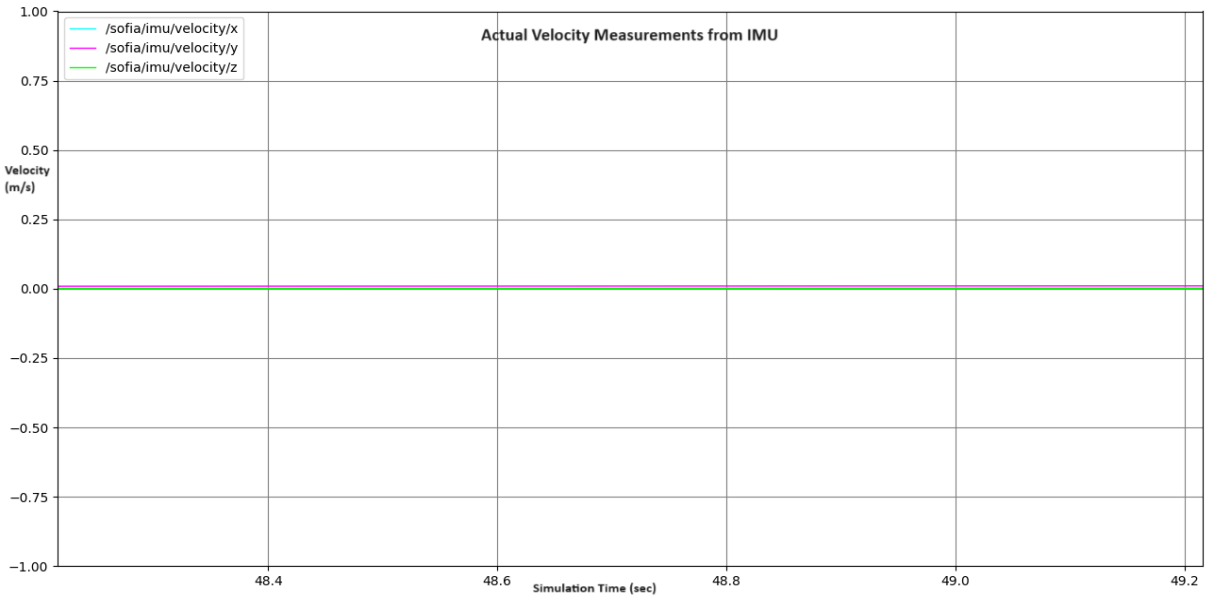

**Εικόνα 6.2-10: Ενοποίηση διαγραμμάτων επιτάχυνσης έπειτα από παρέλευση χρόνου για σενάριο 2**

Από τα παραπάνω, φαίνεται ότι η αστάθεια της επιτάχυνσης διατηρείται ανεξάρτητα από την χρονική στιγμή κατά την οποία θα ληφθεί το στιγμιότυπο. Αυτό σημαίνει ότι η εκκίνηση του συστήματος δεν είναι αυτή που την προκαλεί. Οι μικρές αποκλίσεις της από τις κανονικές τιμές οφείλονται στο γεγονός ότι θα πρέπει να υπάρξει μετέπειτα μετατόπιση του drone που συνεπάγεται συνεχή αλλαγή στις τιμές και μη ικανότητα διατήρησής τους σε μια απόλυτα σταθερή κατάσταση.

Ταυτόχρονα, είναι γνωστό ότι ο αισθητήρας IMU έχει την ικανότητα να μετράει και ταχύτητα σε όλους τους άξονες. Κατά την εκτέλεση του παρόντος σεναρίου, το drone έχει την υποχρέωση να διέρχεται από δύο φάσεις, αυτή της κίνησης και αυτή της ακινητοποίησης. Προσπαθώντας να βρεθούν οι χρονικές στιγμές κατά τις οποίες το drone βρίσκεται σε κάθε μία από αυτές τις καταστάσεις παρατίθενται τα διαγράμματα που φαίνονται στην Εικόνα 6.2-11 και στην Εικόνα 6.2-12. Όπως παρατηρείται στην πρώτη εικόνα, το drone ελαττώνει ταχύτητα κατά τους άξονες x και y γεγονός που φαίνεται να δείχνει ότι προσπαθεί να φτάσει στο επόμενο σημείο της κίνησής του. Την ίδια στιγμή, στην Εικόνα 6.2-12 παρατηρείται το drone σε κατάσταση ακινητοποίησης.

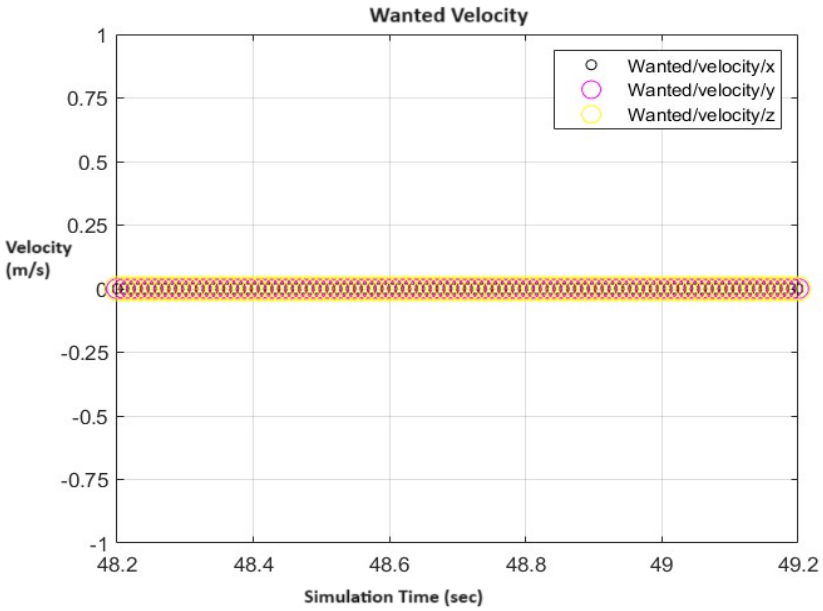



**Εικόνα 6.2-11: Διάγραμμα ταχυτήτων για σενάριο 2 κατά την φάση της επιβράδυνσης**

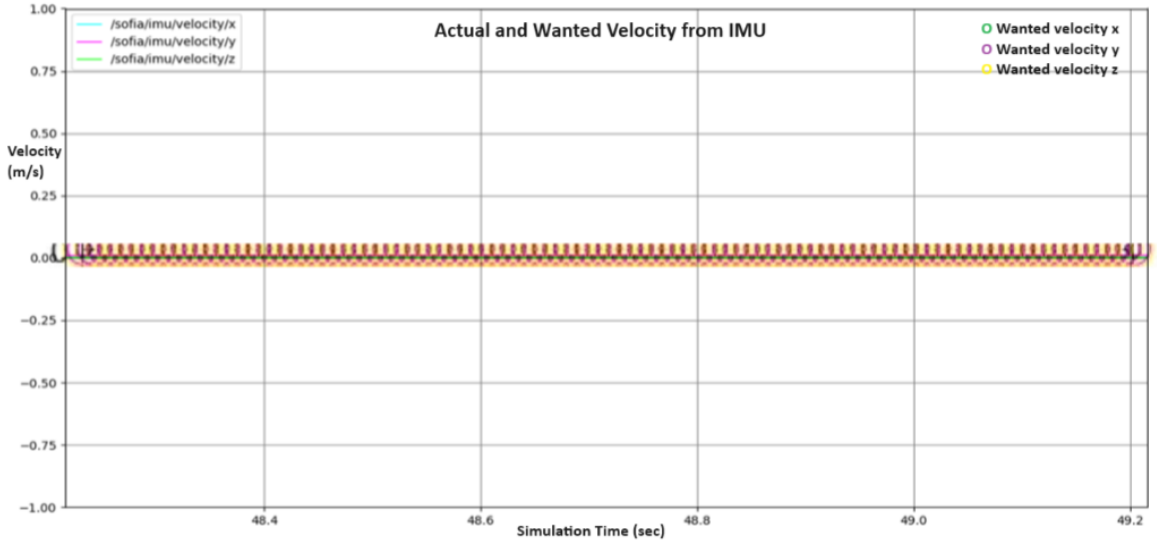



**Εικόνα 6.2-12: Διάγραμμα ταχυτήτων για σενάριο 2 κατά την φάση της ακινητοποίησης**

Για καλύτερη κατανόηση της αντίληψης του αισθητήρα στην Εικόνα 6.2-13 φαίνεται και η επιθυμητή ταχύτητα κατά την διάρκεια της ακινητοποίησης. Τέλος, στην Εικόνα 6.2-14 παρουσιάζεται και το ενοποιημένο διάγραμμα των καταστάσεων για άμεση οπτική σύγκριση.




**Εικόνα 6.2-13: Επιθυμητή κατάσταση ταχύτητας κατά την ακινητοποίηση στο σενάριο 2**




**Εικόνα 6.2-14: Ενοποιημένο διάγραμμα ταχυτήτων για σενάριο 2**

Παρατηρώντας λεπτομερώς τις παραπάνω εικόνες συμπεραίνεται ότι ο αισθητήρας μπορεί να αντιληφθεί ικανοποιητικά όλες τις καταστάσεις του ρομποτικού συστήματος ανεξάρτητα από το ζήτημα της επιτάχυνσης, της επιβράδυνσης ή της ακινητοποίησης. Παράλληλα, εξετάζοντας την σύγκριση μεταξύ επιθυμητής και προσομοιωμένης κατάστασης στην μέτρηση της ταχύτητας γίνεται φανερό ότι οι αποκλίσεις των δύο τιμών είναι μηδενικές.

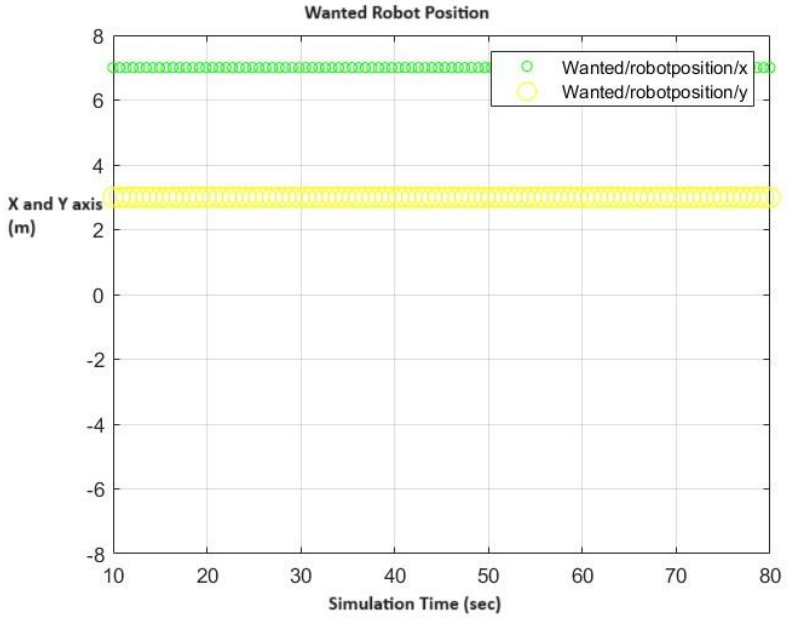
## 6.3 Σενάριο 3: Απλή μετάβαση του drone σε εντελώς τυχαία θέση

Το παρόν σενάριο έχει ως στόχο την μετάβαση από την αρχική θέση σε μία εντελώς τυχαία και την παραμονή σε αυτή. Για την υλοποίηση του σεναρίου η θέση που επιλέχθηκε είναι x=7m, y=3m και z=2m.

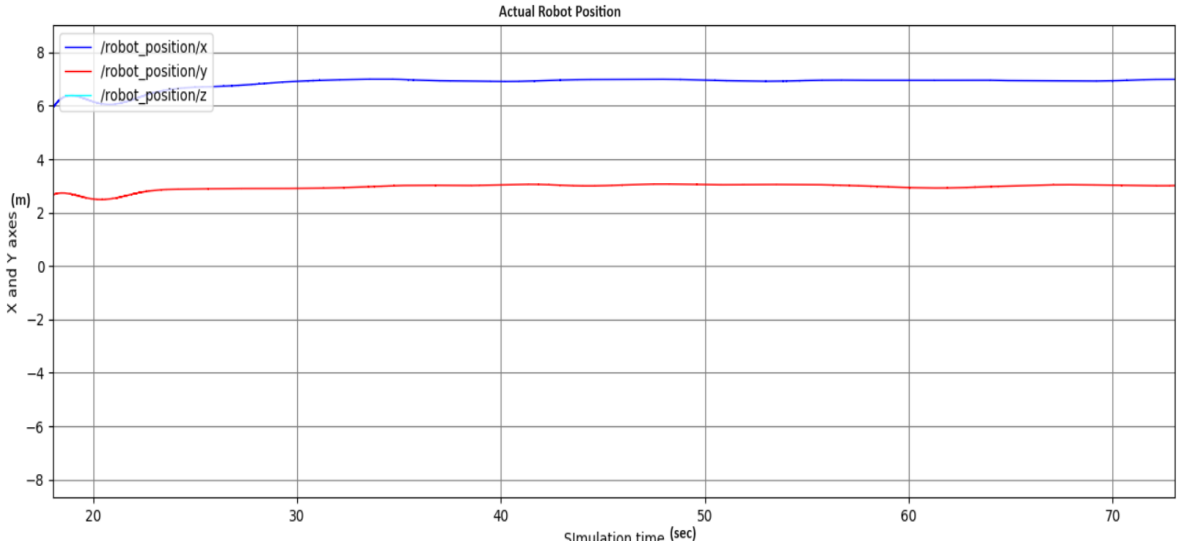
### 6.3.1 Οι μετρήσεις του αισθητήρα UWB κατά την εκτέλεση του σεναρίου 3

Ξεκινώντας την μελέτη του παρόντος σεναρίου είναι σημαντικό να συζητηθεί ο βασικός αισθητήρας UWB. Με σκοπό την ικανοποιητική εξέταση της λειτουργίας του λαμβάνονται διαγράμματα σε στιγμιότυπα που αφορούν αφενός στην εκκίνηση λειτουργίας του σεναρίου και αφετέρου στην ικανότητα του αισθητήρα να λαμβάνει ορθές τιμές με την πάροδο του χρόνου.

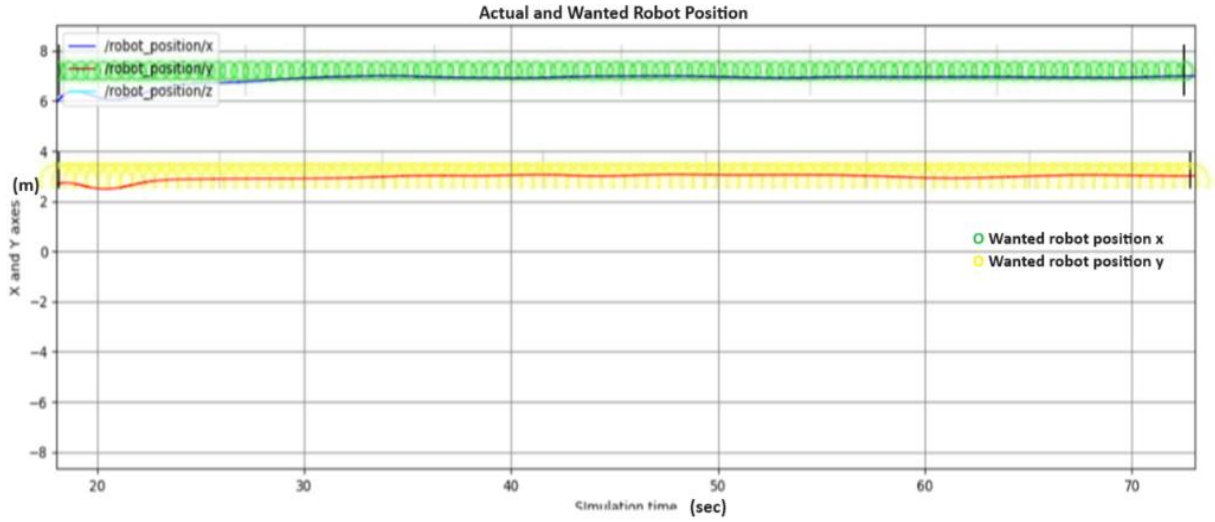
Έχοντας ορίσει την θέση x= 7m και y= 3m η μέτρηση η οποία αναμένεται να δίνει ο αισθητήρας UWB σε κάθε χρονική στιγμή παρουσιάζεται στην Εικόνα 6.3-1 και αποτελεί μια συνεχή γραμμή χωρίς διακυμάνσεις στις καθορισμένες θέσεις. Ωστόσο, κατά την διαδικασία της εκκίνησης το αποτέλεσμα που λαμβάνεται φαίνεται στην Εικόνα 6.3-2 και συνδυαστικά τα δύο διαγράμματα απεικονίζονται στην Εικόνα 6.3-3.




**Εικόνα 6.3-1: Επιθυμητή μέτρηση UWB για σενάριο 3**



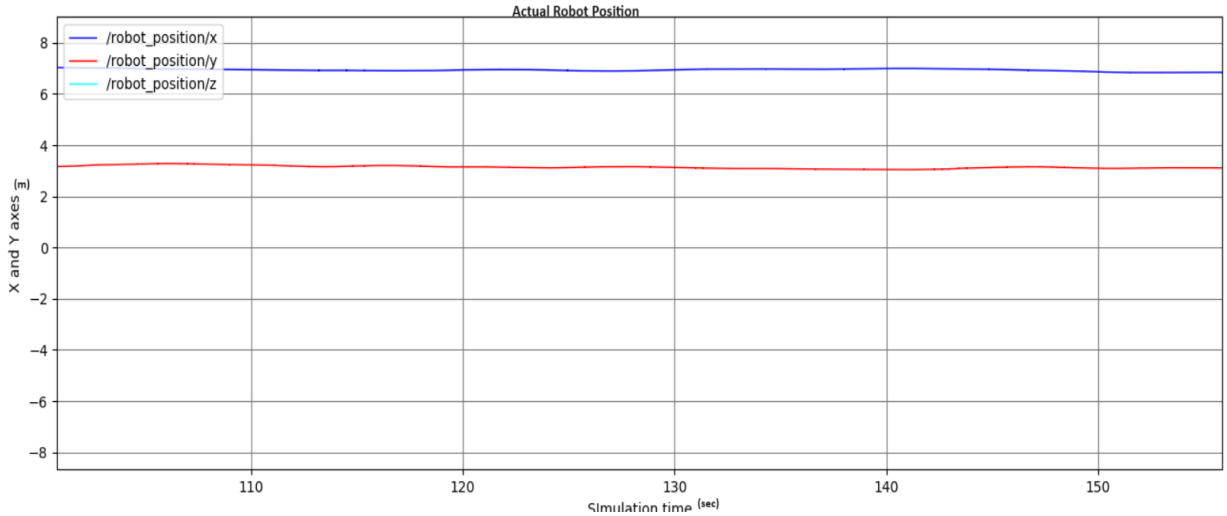

**Εικόνα 6.3-2: Μέτρηση προσομοιωμένου αισθητήρα UWB κατά την εκκίνηση σεναρίου 3**



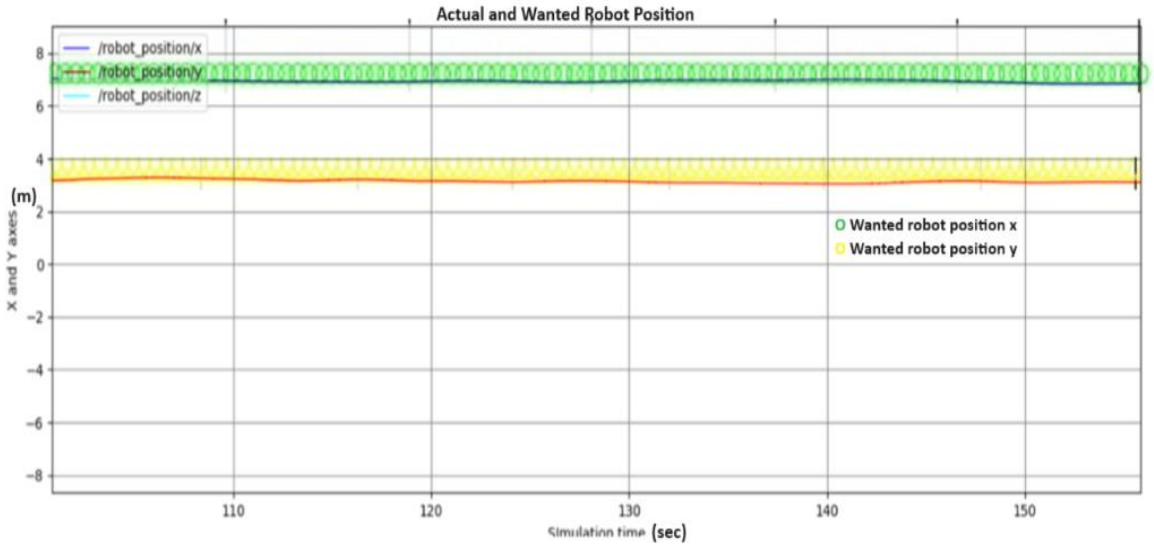

**Εικόνα 6.3-3: Ενοποιημένο διάγραμμα κατά την εκκίνηση αισθητήρα UWB για σενάριο 3**

Μελετώντας με αναλυτικό τρόπο τα παραπάνω παρουσιάζεται αστάθεια κατά την μέτρηση στα αρχικά δευτερόλεπτα της κίνησης. Πιο συγκεκριμένα, αξιοποιώντας την σχέση (6-1) σε κάθε άξονα μπορεί να υπολογιστεί η εν λόγω υπερακόντιση. Αυτή είναι περίπου 10% για τον άξονα y και 11,5% για τον άξονα x. Οι τιμές αυτές δεν είναι σημαντικές ,ωστόσο, καταδεικνύουν ένα πρόβλημα του αισθητήρα να ανταποκριθεί στις ανάγκες του σεναρίου συγκριτικά με τα υπόλοιπα που έχουν μελετηθεί.

Με σκοπό να ελεγχθεί εάν αυτό το πρόβλημα παραμένει σταθερό ή παρατηρείται μόνο κατά την διάρκεια της εκκίνησης λαμβάνεται άλλο ένα στιγμιότυπο έπειτα από την πάροδο αρκετού χρόνου. Αυτό φαίνεται στην Εικόνα 6.3-4. Ειδικότερα, έχοντας γνώση της επιθυμητής μέτρησης του αισθητήρα η οποία είναι αυτή που εντοπίζεται στην Εικόνα 6.3-1 και παρατηρώντας το ενοποιημένο διάγραμμα στην Εικόνα 6.3-5 γίνεται αντιληπτό ότι το ζήτημα της αστάθειας εξαλείφεται. Ωστόσο, και πάλι οι μετρήσεις εμφανίζουν μικρές διακυμάνσεις που δεν είναι σημαντικές.

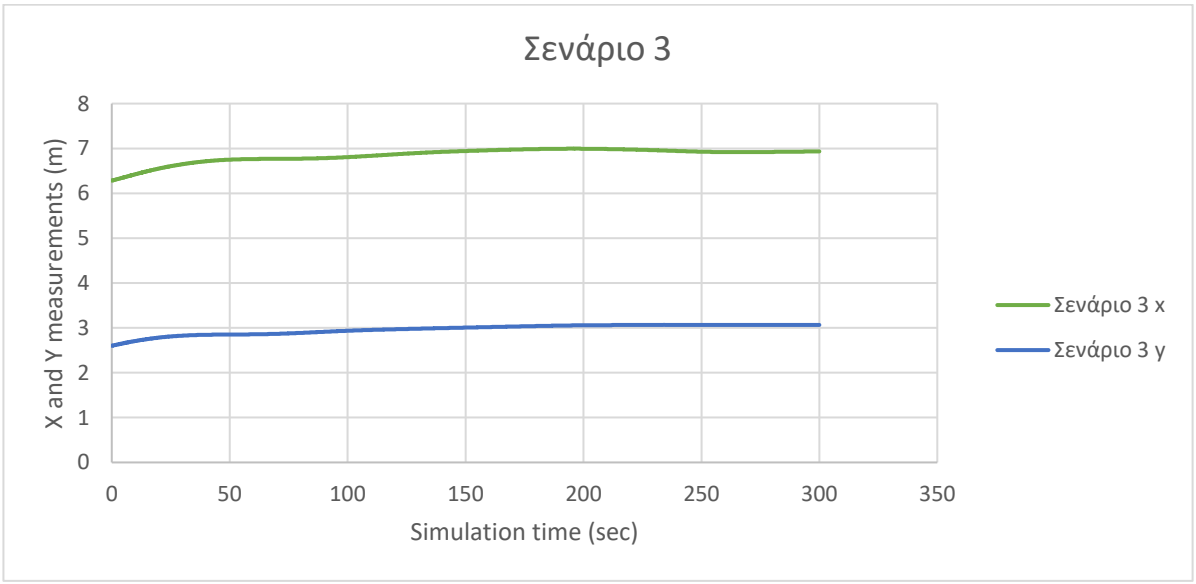



**Εικόνα 6.3-4: Προσομοιωμένος αισθητήρας UWB μετά από χρόνο κατά το σενάριο 3**



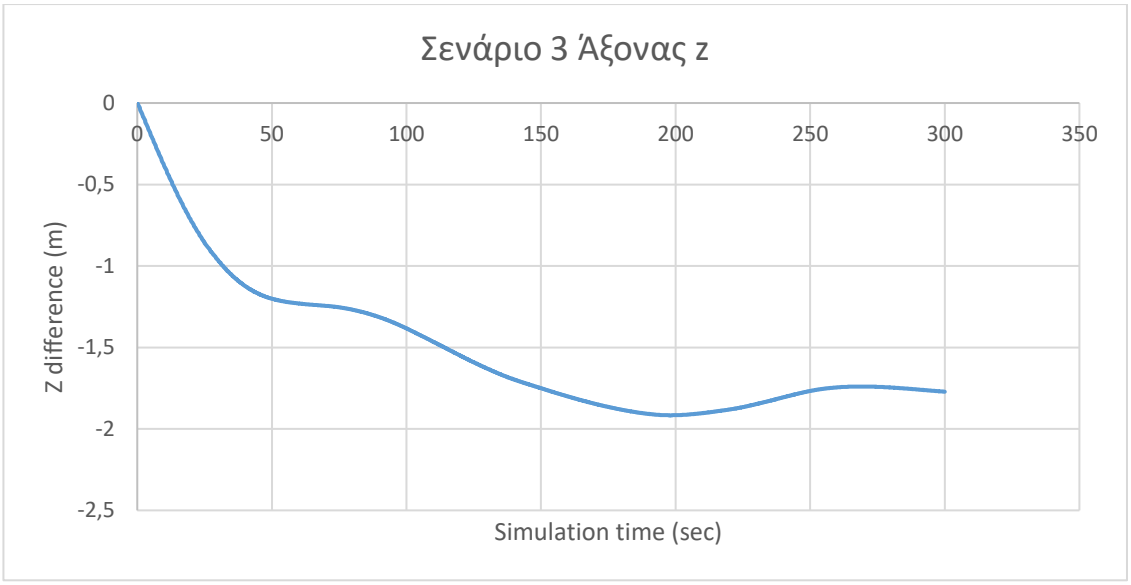

**Εικόνα 6.3-5: Ενοποιημένο διάγραμμα UWB μετά από παρέλευση χρόνου για σενάριο 3**

Με στόχο την κατανόηση των μετρήσεων που παρέχει ο αισθητήρας πρέπει να δοθεί διάγραμμα με τον υπολογισμό της θέσης του drone κατά την εκτέλεση της κίνησής του. Το διάγραμμα αυτό φαίνεται στην Εικόνα 6.3-6 και αποδεικνύει ότι ο υπολογισμός της θέσης του εναέριου οχήματος στους άξονες x και y γίνεται με ακρίβεια και με μηδενικές διακυμάνσεις.




**Εικόνα 6.3-6: Διάγραμμα συνολικής κίνησης για σενάριο 3**

Κρίσιμο ζήτημα θεωρείται αυτό του υπολογισμού της θέσης του drone στον άξονα z. Πιο συγκεκριμένα, αυτό δεν μπορεί να γίνει από τον κώδικα καθώς οι μεμονωμένες τιμές που προκύπτουν είναι ανακριβείς και χωρίς νόημα. Εξαιτίας αυτού, προτιμάται να παρουσιαστεί ένα διάγραμμα το οποίο θα απεικονίζει τις διαφορές της εκάστοτε θέσης με την προηγούμενή της. Αυτό φαίνεται στην

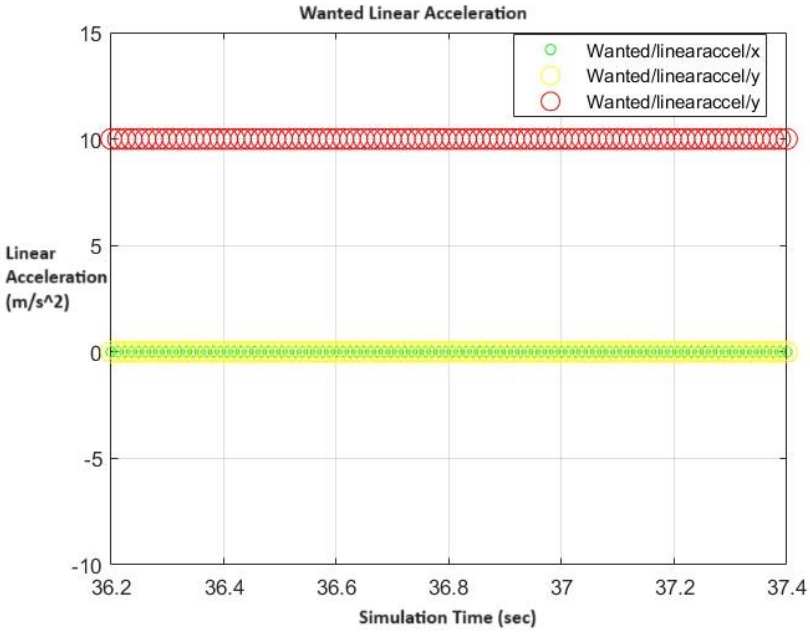



**Εικόνα 6.3-7: Διαφορές των τιμών των μετρήσεων στον άξονα z του σεναρίου 3**

Σε αυτό το σενάριο παρατηρείται ότι οι διαφορές αυξάνονται με την πάροδο του χρόνου. Αυτό σημαίνει ότι ο αισθητήρας αφενός δεν καταγράφει σωστά την υψομετρική τιμή και αφετέρου δεν την διατηρεί σταθερή. Αυτό σηματοδοτεί μια μη ορθή μέτρηση καθώς σε αυτό το σενάριο το drone λαμβάνει ένα τελικό ύψος και παραμένει σε αυτό σε όλη την διάρκεια της κίνησής του.

### 6.3.2 Οι μετρήσεις του αισθητήρα IMU για το σενάριο 3

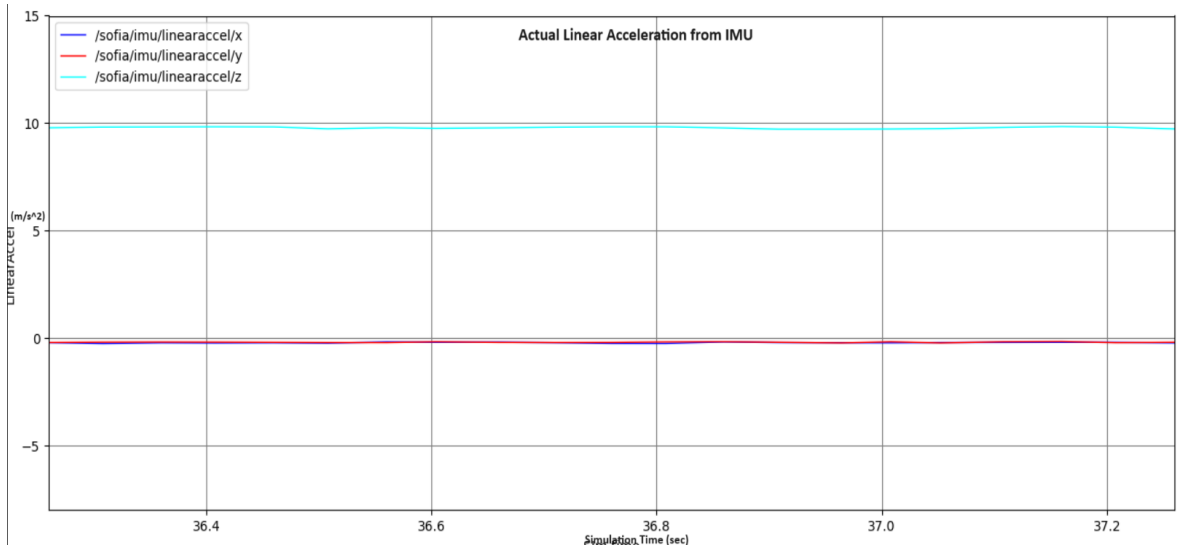
Κατά την διάρκεια εκτέλεσης του σεναρίου η κίνηση του drone δύναται να χωριστεί σε τρείς καταστάσεις: την επιτάχυνση, την επιβράδυνση και την σταθερή κατάσταση. Έχοντας αυτόν τον διαχωρισμό υπόψη η γραμμική επιτάχυνση η οποία υπολογίζεται από τον αισθητήρα κατά την διάρκεια της στασιμότητας θα πρέπει να συμμορφώνεται πλήρως με τις επιθυμητές τιμές. Αυτές φαίνονται στην Εικόνα 6.3-8 και ισχύουν για οποιαδήποτε χρονική στιγμή κατά την οποία το drone παραμένει σταθερό στην τυχαία του θέση.



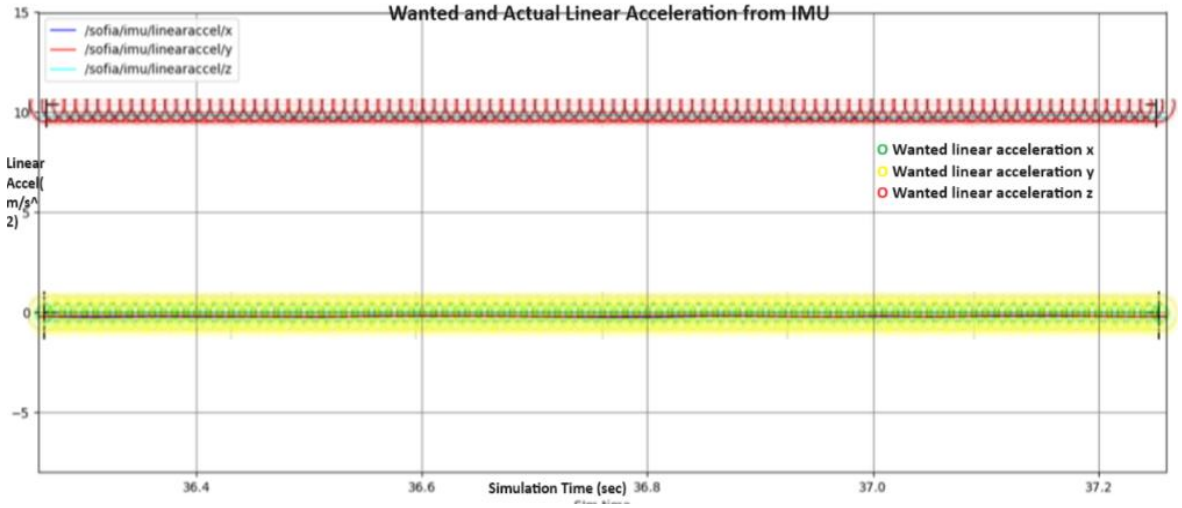

**Εικόνα 6.3-8: Επιθυμητή γραμμική επιτάχυνση κατά την στασιμότητα για σενάριο 3**

Κατά την διαδικασία της εκκίνησης της προσομοίωσης λαμβάνεται το στιγμιότυπο που φαίνεται στην Εικόνα 6.3-9. Από το ενοποιημένο διάγραμμα στην Εικόνα 6.3-10 φαίνεται ότι οι μετρήσεις του αισθητήρα IMU για

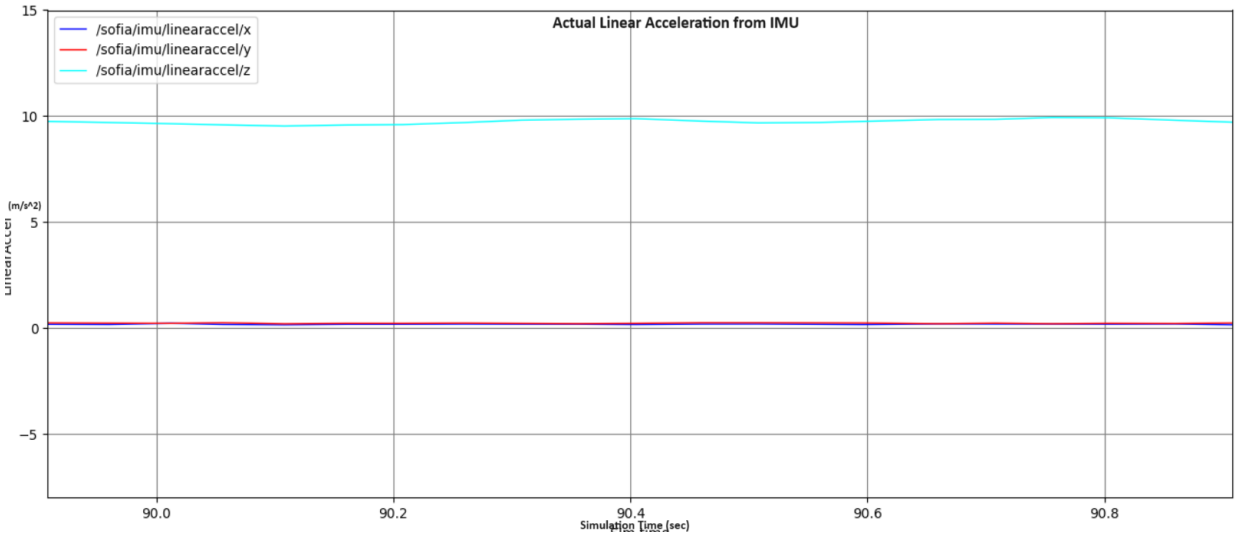
την γραμμική επιτάχυνση είναι σταθερές και ακριβείς. Ωστόσο, κατά την διάρκεια της προσομοίωσης παρατηρείται ότι η επιτάχυνση η οποία υπολογίζεται στον άξονα z εμφανίζει διακυμάνσεις. Αυτό φαίνεται στην Εικόνα 6.3-11. Αυτές δεν είναι σοβαρές με την μεγαλύτερη να φτάνει το 2%. Παρόλα αυτά καταδεικνύουν μικρές αστάθειες στην λειτουργία του αισθητήρα κατά την εκτέλεση αυτού του σεναρίου.




**Εικόνα 6.3-9: Επιτάχυνση κατά την εκκίνηση του σεναρίου 3**

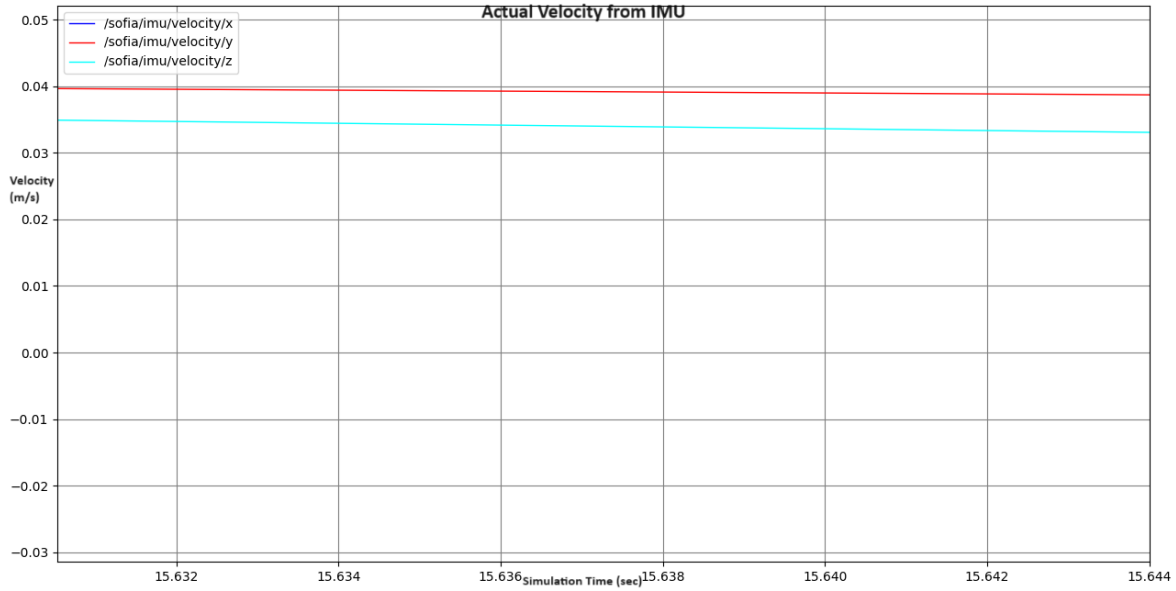



**Εικόνα 6.3-10: Ενοποιημένο διάγραμμα επιτάχυνσης για σενάριο 3**



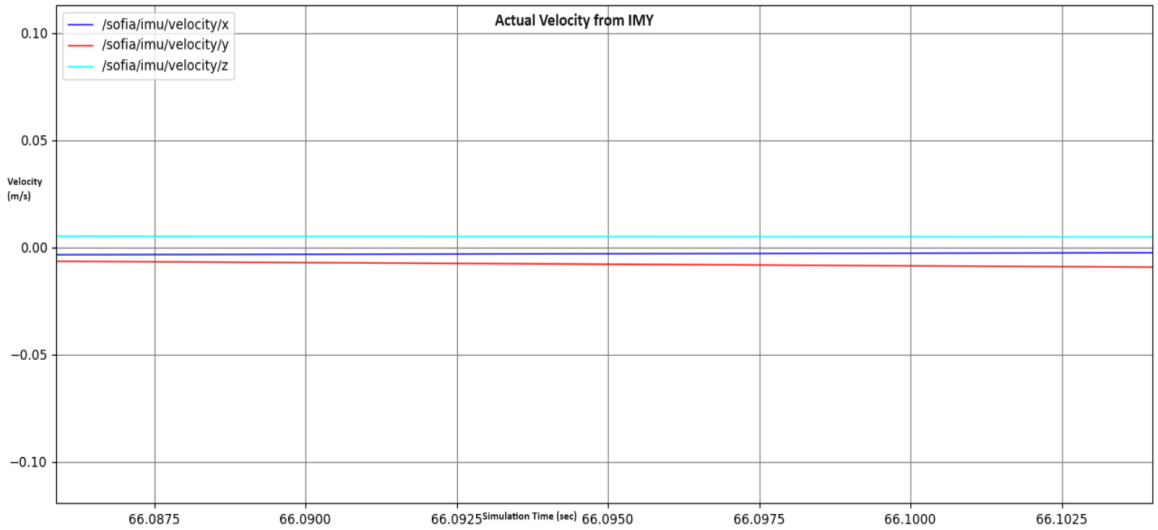

**Εικόνα 6.3-11: Διάγραμμα επιτάχυνσης κατά την παρέλευση χρόνου στο σενάριο 3**

Την ίδια στιγμή, άξια αναφοράς είναι η ταχύτητα του μη επανδρωμένου αεροσκάφους. Πιο αναλυτικά, προσπαθώντας να εξετασθεί εάν ο αισθητήρας έχει την ικανότητα να ακολουθήσει την φάση της επιτάχυνσης του συστήματος λαμβάνεται το στιγμιότυπα που φαίνεται στην Εικόνα 6.3-12. Από αυτό προκύπτει ότι ο αισθητήρας IMU δύναται να ακολουθεί με αξιόπιστο τρόπο τις μεταβολές της ταχύτητας του αεροσκάφους. Παράλληλα, η καθοδική τάση της ταχύτητας που παρατηρείται αποδεικνύει ακόμα πιο τρανά αυτό το συμπέρασμα καθώς δύναται να γίνει αντιληπτή και η κατάσταση της επιβράδυνσης.

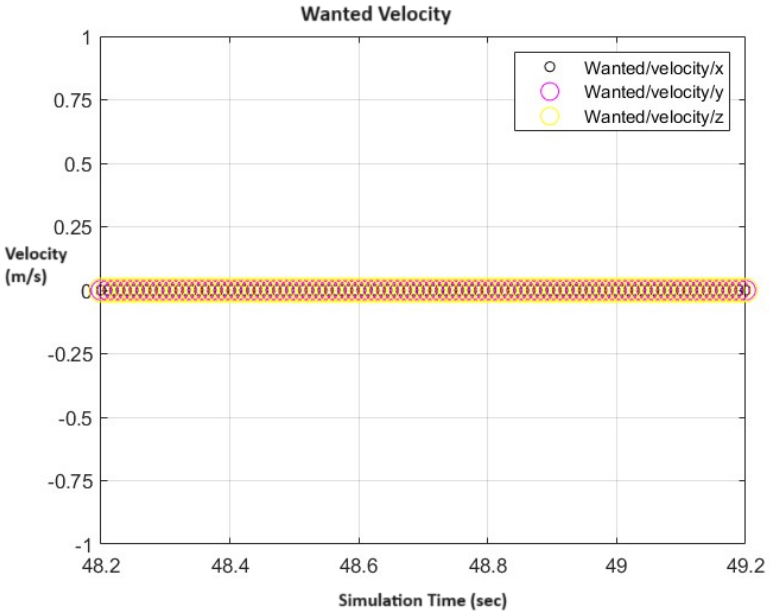



**Εικόνα 6.3-12: Μέτρηση ταχύτητας κατά την επιτάχυνση στο σενάριο 3**

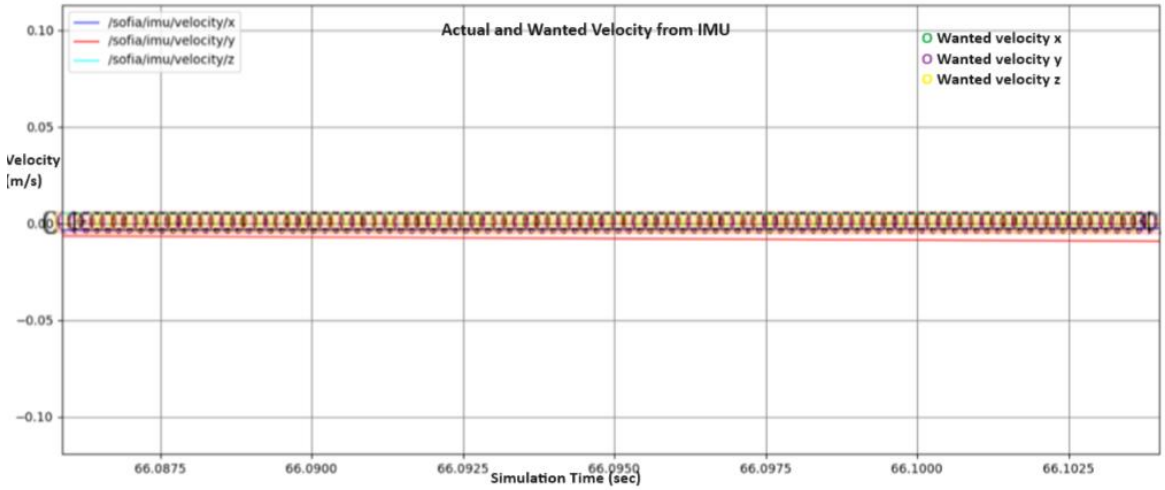
Ακόμα, σημαντική είναι και η εξέταση της ευστάθειας των μετρήσεων του συστήματος IMU. Ειδικότερα, για τον έλεγχο αυτό προτιμάται ένα στιγμιότυπο κατά το οποίο το drone είναι σίγουρα σταθερό στην τελική του θέση. Το στιγμιότυπο αυτό παρατηρείται στην Εικόνα 6.3-13. Όπως είναι λογικό, η επιθυμητή μέτρηση του αισθητήρα είναι η μηδενική θέση και φαίνεται στην Εικόνα 6.3-14. Παρατηρώντας τόσο τις δύο εικόνες όσο και το ενοποιημένο διάγραμμα στην Εικόνα 6.3-15 γίνεται αντιληπτό ότι ο αισθητήρας αποκλίνει ελάχιστα από το απόλυτο μηδέν με μία απόκλιση της τάξεως του 0,01. Η τιμή αυτή χαρακτηρίζεται ασήμαντη και δεν προσδίδει ανακρίβειες στον αισθητήρα.




**Εικόνα 6.3-13: Ταχύτητα κατά την σταθερότητα του drone στο σενάριο 3**




**Εικόνα 6.3-14: Επιθυμητή ταχύτητα σταθερής κατάστασης στο σενάριο 3**



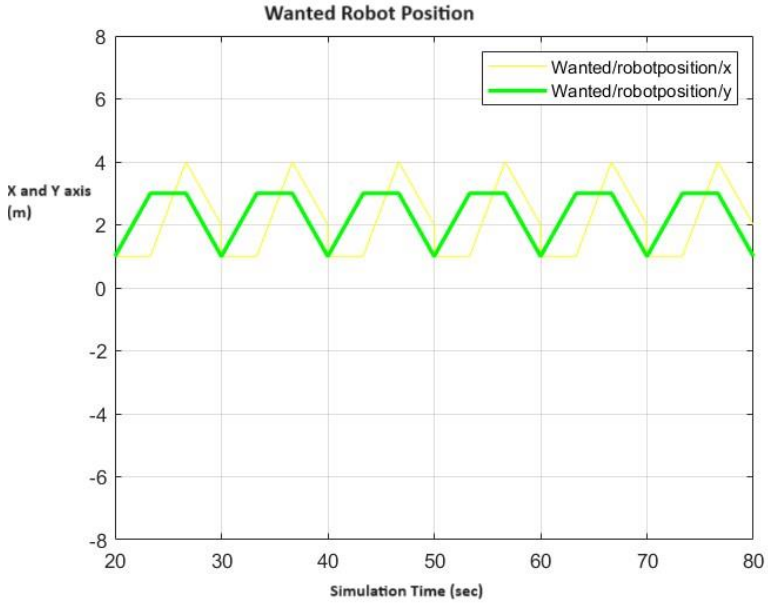

**Εικόνα 6.3-15: Ενοποιημένο διάγραμμα ταχύτητας σταθερής κατάστασης στο σενάριο 3**

## 6.4 Σενάριο 4: Σύνθετη τετραγωνική τροχιά υπό σταθερό ύψος.

Σκοπός του παρόντος σεναρίου είναι η μελέτη μιας περισσότερο σύνθετης τροχιάς. Αναλυτικότερα, η κίνηση που εκτελείται περιλαμβάνει τέσσερα σημεία τα οποία δύναται να σχηματίσουν τετραγωνικό σχήμα. Πιο συγκεκριμένα, το drone ξεκινά από το σημείο (1,1,2) οδηγείται στο σημείο (3,1,2). Έπειτα φτάνει στο (3,4,2) και τελικά καταλήγει στο σημείο (1,4,2). Αυτή η κίνηση είναι περιοδική.

### 6.4.1 Ο αισθητήρας UWB στο σενάριο 4.

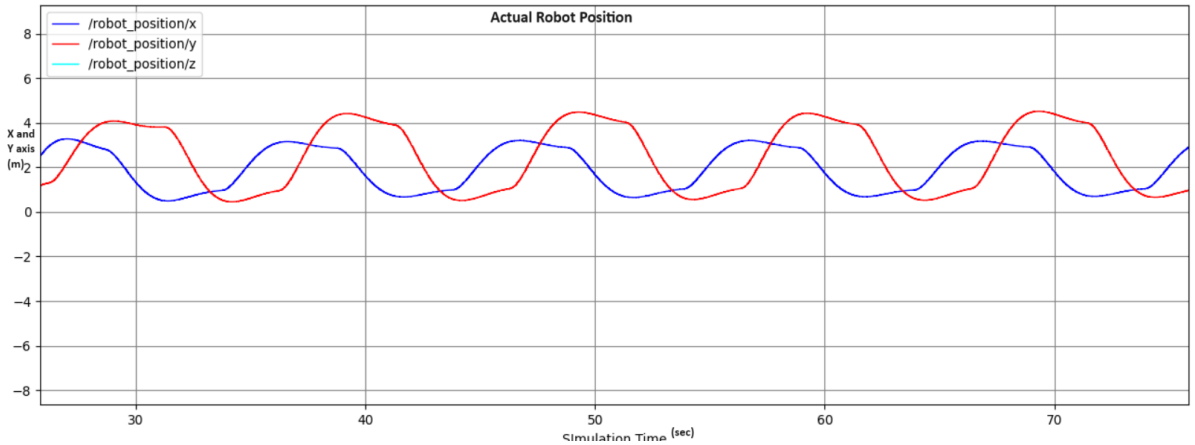
Ξεκινώντας την ανάλυση του τέταρτου κατά σειρά σεναρίου είναι σημαντικό να ερευνηθεί η ανταπόκριση του κώδικα που έχει γραφτεί για τον βασικό αισθητήρα UWB. Πιο αναλυτικά, η τροχιά η οποία προσομοιώνεται είναι τετραγωνική και περιοδική. Αυτή, σε επίπεδο διαγράμματος φαίνεται στην Εικόνα 6.4-1.




**Εικόνα 6.4-1: Επιθυμητή τροχιά από UWB για σενάριο 4**

Είναι προφανές το γεγονός ότι κατά την κίνηση παρατηρούνται μεταβολές τόσο στον άξονα των x όσο και σε αυτόν των y. Ωστόσο, οι μεταβολές αυτές στην επιθυμητή κατάσταση είναι απότομες και γραμμικές όπως καταδεικνύουν οι κλίσεις του παραπάνω διαγράμματος. Αυτό, δεν μπορεί να αποτυπωθεί στην προσομοίωση καθώς όπως είναι λογικό και στον φυσικό κόσμο οι αλλαγές θέσεων είναι πιο ομαλές και ήπιες. Επομένως,

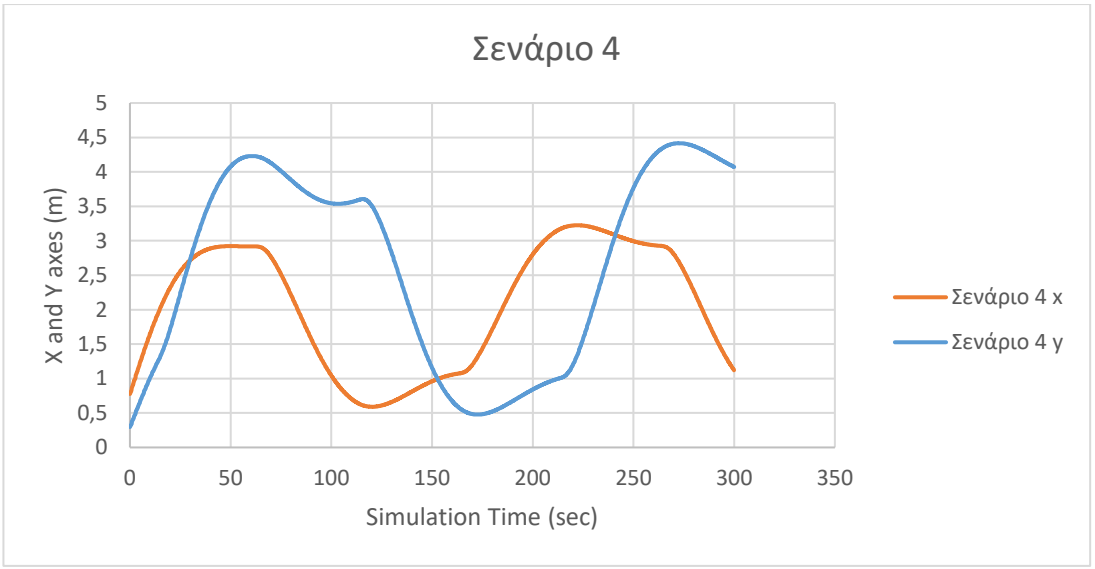
το αποτέλεσμα της προσομοίωσης δίνεται στην Εικόνα 6.4-2. Την ίδια στιγμή, αποτυπώνεται και το ενοποιημένο διάγραμμα στην Εικόνα 6.4-3 κατά το οποίο φαίνεται λεπτομερώς ότι ο αισθητήρας UWB μπορεί να εντοπίζει με σημαντική ακρίβεια την θέση του drone κατά την κίνηση του.




**Εικόνα 6.4-2:Τροχιά από κώδικα UWB για σενάριο 4**

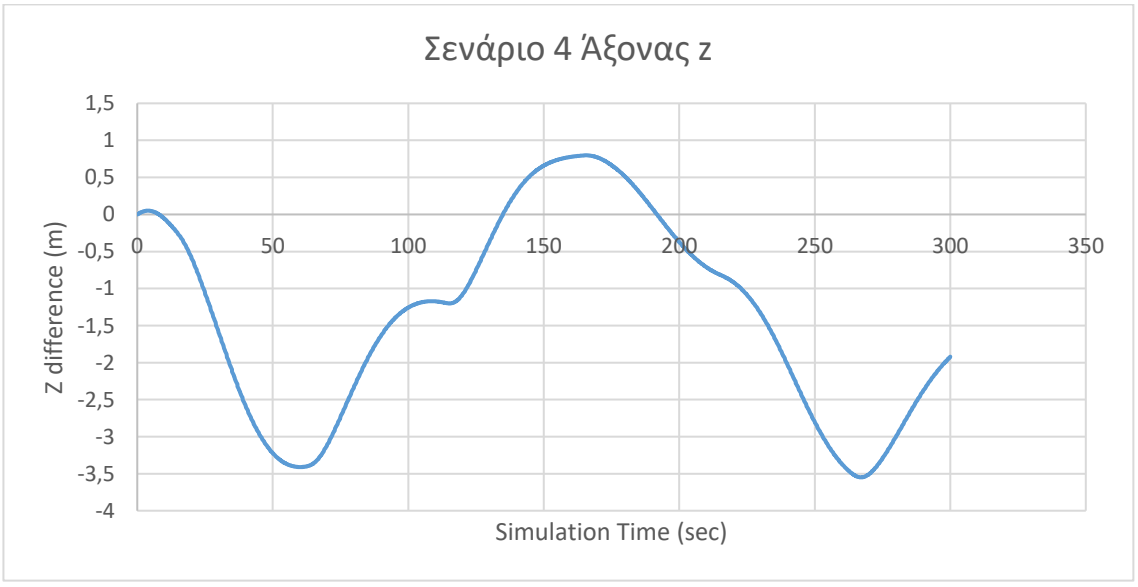

**Εικόνα 6.4-3: Ενοποίηση διαγραμμάτων UWB για σενάριο 4**

Για καλύτερη αντίληψη των μετρήσεων παρέχεται η συνολική κίνηση του drone όπως αυτή καταγράφεται από τον προσομοιωμένο αισθητήρα UWB. Αυτή παρουσιάζεται στην Εικόνα 6.4-4. Αναλυτικότερα, μπορεί να παρατηρηθεί ότι η τροχιά του drone καταγράφεται με επιτυχία χωρίς διακυμάνσεις και ανακρίβειες. Η μοναδική διαφορά με την επιθυμητή τροχιά στην Εικόνα 6.4-1 αποτελούν οι στρογγυλοποιημένες κορυφές οι οποίες είναι φυσιολογικές στον πραγματικό κόσμο καθώς οι απότομες μεταβολές κινήσεων αποφεύγονται.




**Εικόνα 6.4-4: Συνολική κίνηση του drone στο σενάριο 4**

Βασικό είναι να παρουσιαστούν και οι υψομετρικές μετρήσεις του αισθητήρα UWB. Αυτές δεν είναι ακριβείς και δεν μπορούν να υπολογίσουν το ύψος στο οποίο βρίσκεται το drone. Εξαιτίας αυτού, προτιμάται να δειχθεί η διαφορά της κάθε θέσης με την προηγούμενή της με στόχο την κατανόηση της αντίληψης του ύψους από τον αισθητήρα. Αυτό παρέχεται στην Εικόνα 6.4-5.



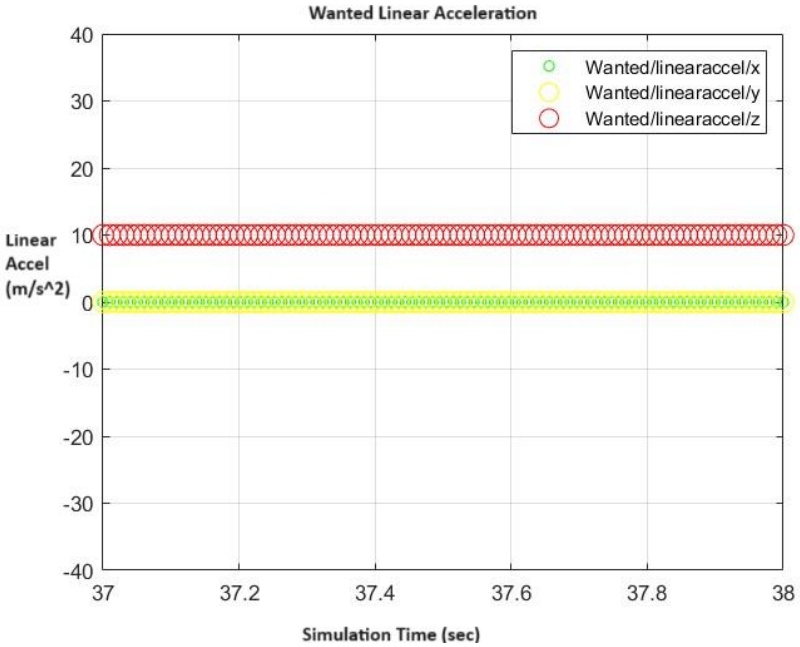

**Εικόνα 6.4-5: Υψομετρικές διαφορές των μετρήσεων στο σενάριο 4**

Από το παραπάνω διάγραμμα παρατηρείται ότι η διαφορά των υψών μεταξύ των θέσεων δεν είναι σταθερή. Αυτό σημαίνει ότι ο αισθητήρας δεν παρέχει την ίδια τιμή κάθε χρονική στιγμή. Παράλληλα, οι αποστάσεις των διαφορών από το μηδέν είναι σημαντικές για ορισμένες χρονικές στιγμές γεγονός που αποδεικνύει αναξιοπιστία στους υπολογισμούς και στις μετρήσεις.

### 6.4.2 Ο αισθητήρας IMU στο σενάριο 4

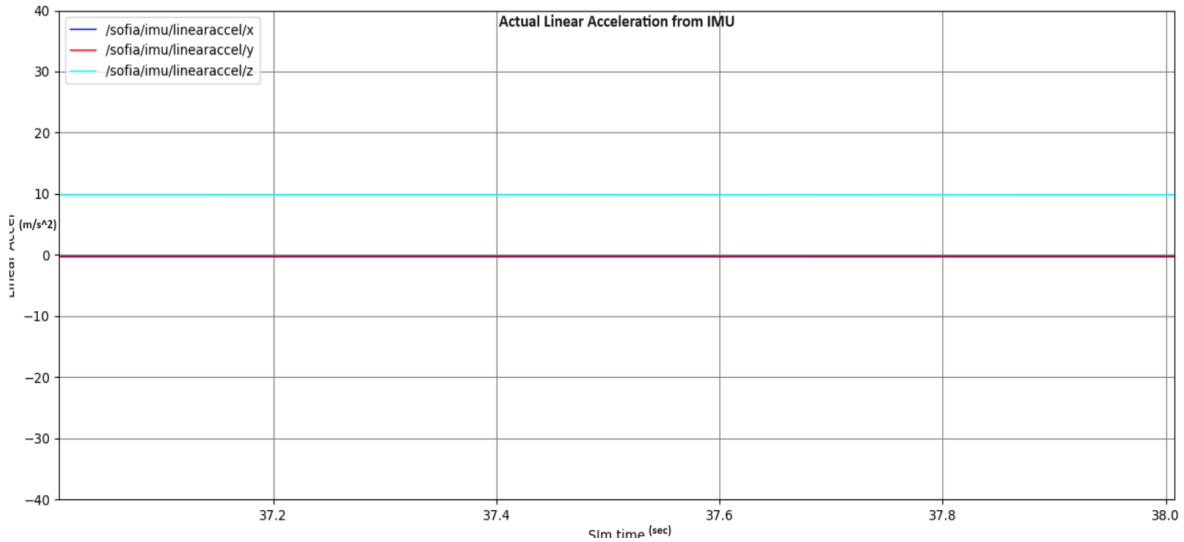
Με στόχο τον πλήρη έλεγχο της κατάστασης του drone θα πρέπει να υπολογίζεται με ακρίβεια τόσο η γραμμική επιτάχυνση των αξόνων όσο και η ταχύτητα αυτού. Πιο αναλυτικά, εφόσον το drone αλλάζει θέσεις στον χώρο, διαθέτει μεταβολές κατά την κίνησή του. Αυτές είναι η επιτάχυνση, η επιβράδυνση και η σταθερή κατάσταση.

Ξεκινώντας την αναφορά από την μελέτη την γραμμικής επιτάχυνσης, αυτή θα πρέπει να διατηρείται σταθερή κατά την μετάβαση του drone στις θέσεις του. Ταυτόχρονα, θα πρέπει να έχει τις καθορισμένες τιμές της προσομοίωσης οι οποίες φαίνονται στην Εικόνα 6.4-6.




**Εικόνα 6.4-6: Επιθυμητή γραμμική επιτάχυνση για σενάριο 4**

Λαμβάνοντας στιγμιότυπο μια χρονική στιγμή κατά την οποία το drone έχει φτάσει σε σταθερή θέση παρέχεται το αποτέλεσμα που φαίνεται στην Εικόνα 6.4-7. Την ίδια στιγμή, για καλύτερη σύγκριση το ενοποιημένο διάγραμμα των δύο καταστάσεων παρουσιάζεται στην Εικόνα 6.4-8.

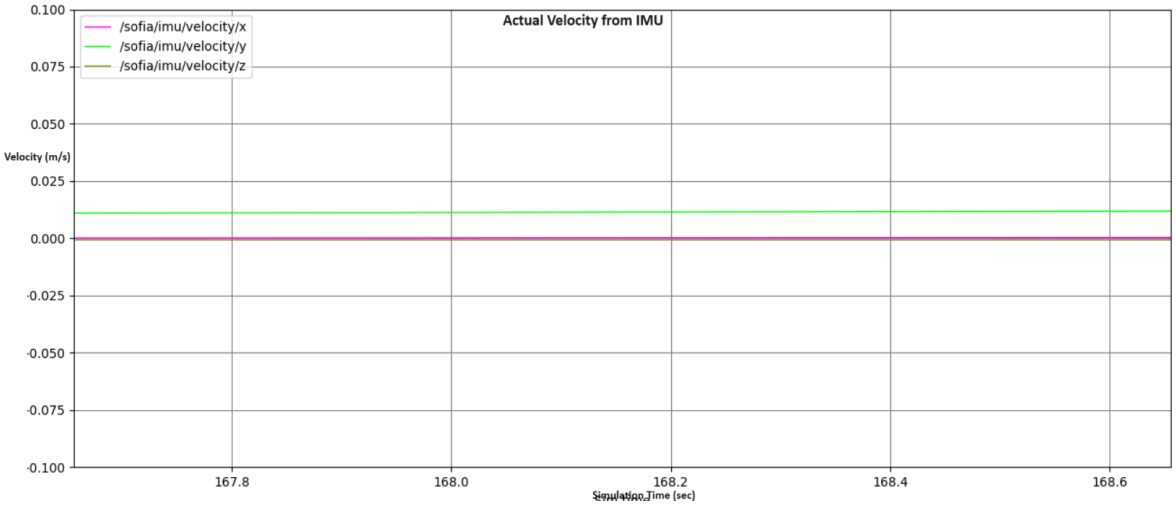



**Εικόνα 6.4-7: Γραμμική επιτάχυνση κατά την προσομοίωση στο σενάριο 4**


**Εικόνα 6.4-8: Ενοποιημένο διάγραμμα επιτάχυνσης στο σενάριο 4**

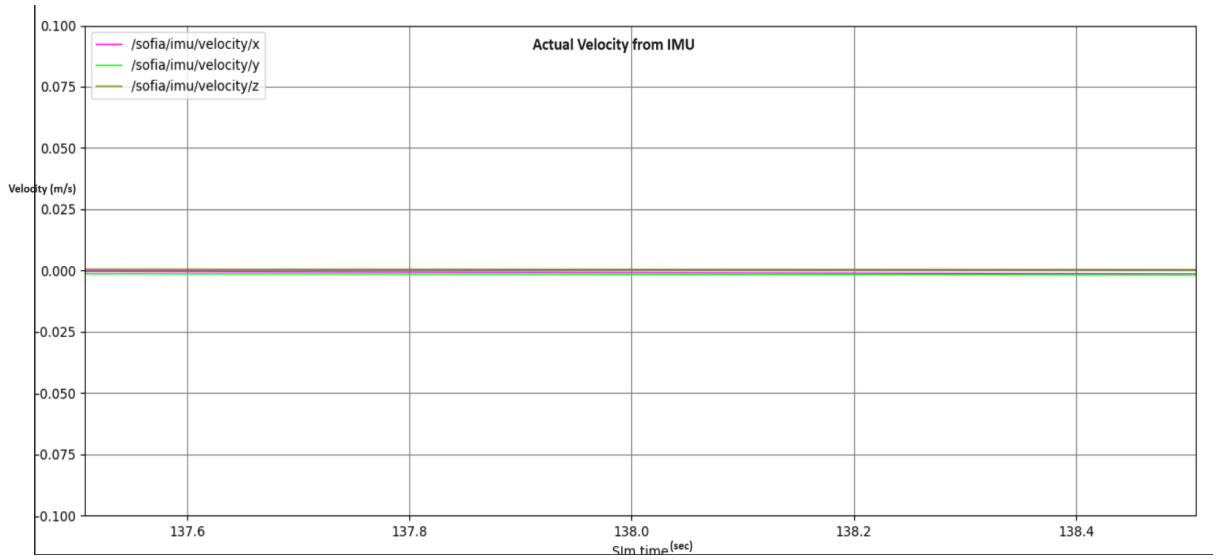
Ο αισθητήρας μετράει ικανοποιητικά την γραμμική επιτάχυνση του συστήματος σε κάθε άξονα και καταδεικνύει τις επιθυμητές τιμές χωρίς αποκλίσεις. Την ίδια στιγμή, αξίζει να σημειωθεί ότι οι χρονικές στιγμές της επιτάχυνσης ή της επιβράδυνσης είναι δύσκολο να ληφθούν ως στιγμιότυπα και για αυτό το λόγο δεν καταδεικνύονται σε αυτό το επίπεδο αλλά φαίνονται στα διαγράμματα των ταχυτήτων.

Ολοκληρώνοντας το ζήτημα της μέτρησης της γραμμικής επιτάχυνσης ερευνάται αυτό της ταχύτητας. Πιο αναλυτικά, είναι σημαντικό για την προσομοίωση να βρεθεί εάν ο αισθητήρας έχει την ικανότητα να ακολουθεί όλες τις μεταβολές της κίνησης του συστήματος. Όπως εικονίζεται στην Εικόνα 6.4-9 ο αισθητήρας IMU παρακολουθεί με απόλυτη ακρίβεια μια τυχαία κατάσταση επιτάχυνσης με αποτέλεσμα αυτό το ερώτημα να λαμβάνει απάντηση.

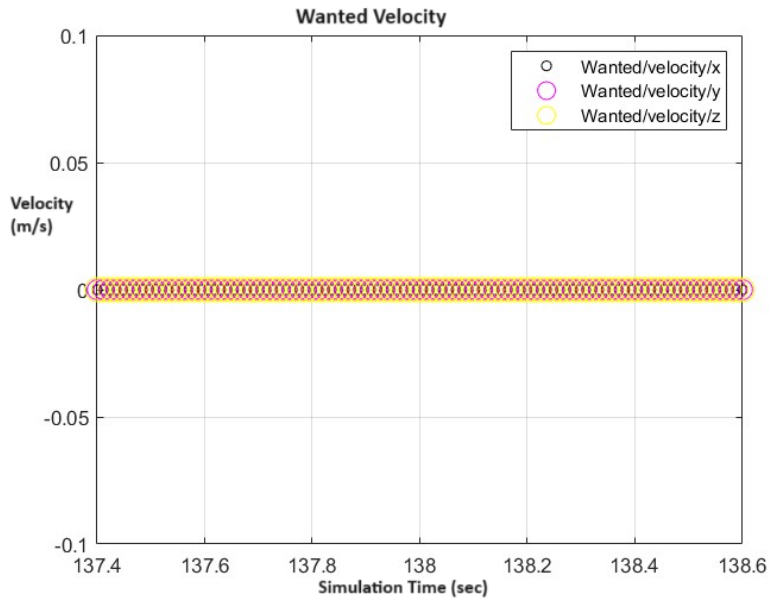



**Εικόνα 6.4-9: Κατάσταση επιτάχυνσης κατά την προσομοίωση στο σενάριο 4**

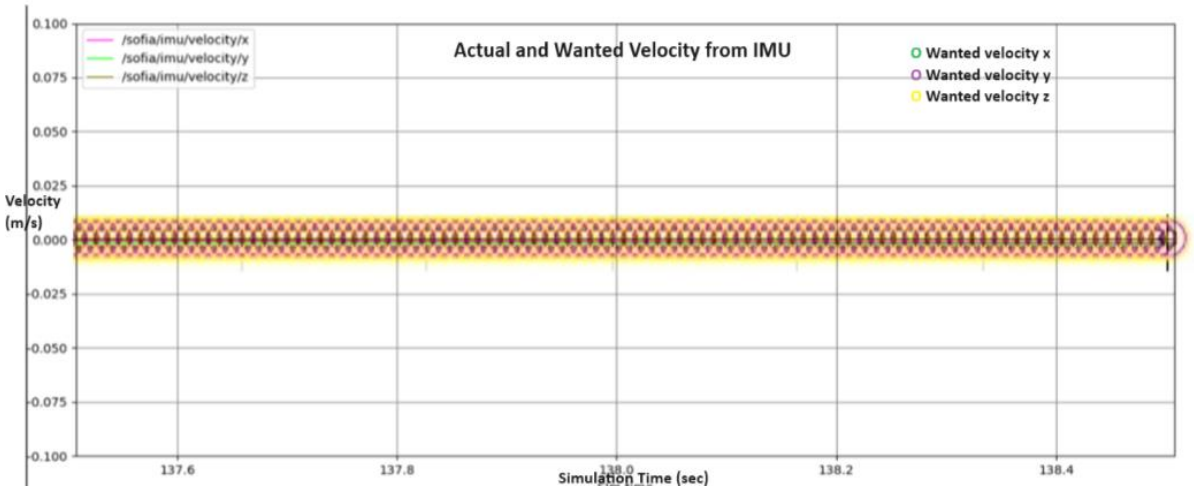
Ταυτόχρονα, είναι σημαντικό να μην παρουσιάζονται αποκλίσεις κατά τις μετρήσεις της ταχύτητας. Για να μελετηθεί αυτό, λαμβάνεται στιγμιότυπο κατά το οποίο το drone είναι γνωστό πως βρίσκεται ακίνητο σε σταθερή θέση. Η μέτρηση που λαμβάνεται από τον αισθητήρα φαίνεται στην Εικόνα 6.4-10 ενώ η επιθυμητή τιμή στην Εικόνα 6.4-11. Όπως παρατηρείται και στο ενοποιημένο διάγραμμα στην Εικόνα 6.4-12 η μέτρηση δεν αποκλίνει καθόλου από την αναμενόμενη τιμή. Το γεγονός αυτό καθιστά τον αισθητήρα ακριβή.




**Εικόνα 6.4-10: Ταχύτητα κατά την ακινητοποίηση του drone στο σενάριο 4**




**Εικόνα 6.4-11: Επιθυμητή ταχύτητα ακινητοποίησης στο σενάριο 4**




**Εικόνα 6.4-12: Ενοποιημένο διάγραμμα ταχύτητας ακινητοποίησης στο σενάριο 4**

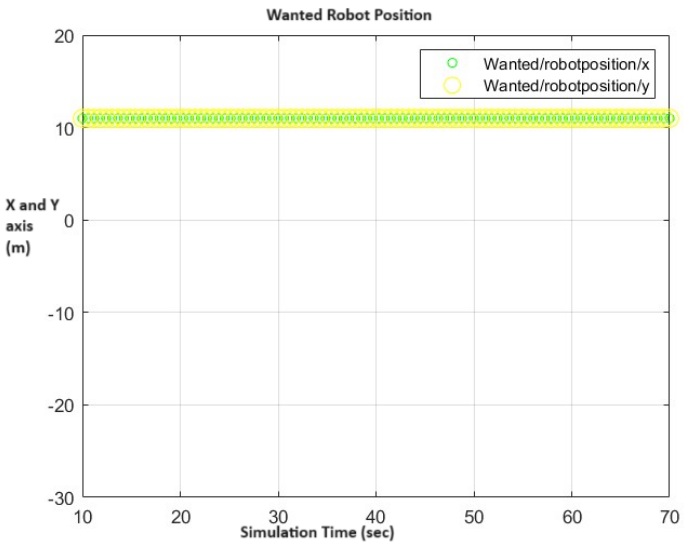
## 6.5 Σενάριο 5: Μετάβαση σε θέση η οποία βρίσκεται εκτός των αγκυρών.

Με στόχο την εξέταση της πλήρους λειτουργίας του κώδικα θεωρήθηκε σημαντικό να μελετηθεί μια θέση η οποία δεν θα βρίσκεται εντός των ορίων που καθορίζουν οι γνωστοί αισθητήρες των αγκυρών. Η επιλογή αυτή έγινε με σκοπό τον έλεγχο του συστήματος σε μία κατάσταση η οποία δεν είναι αναμενόμενη και επιθυμητή αλλά μπορεί να εμφανιστεί υπό κανονικές συνθήκες. Εξαιτίας αυτού, η θέση που επιλέχθηκε είναι x=11m, y=11m και z=2m η οποία δεν απέχει πολύ από τις γνωστές θέσεις καθώς αυτές βρίσκονται στα x=10m και y=10m.

### 6.5.1 Η συμπεριφορά του αισθητήρα UWB κατά το σενάριο 5

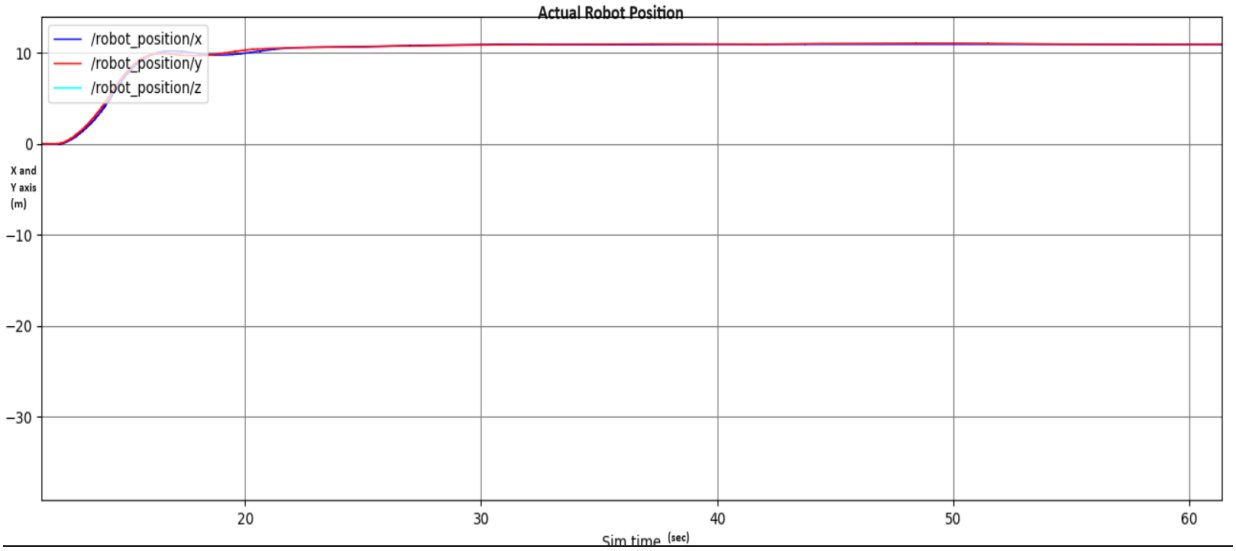
Ξεκινώντας την μελέτη του παρόντος σεναρίου είναι σημαντικό να συζητηθεί η συμπεριφορά του αισθητήρα UWB. Με σκοπό την ικανοποιητική εξέταση της λειτουργίας του λαμβάνονται διαγράμματα σε στιγμιότυπα που αφορούν αφενός στην εκκίνηση λειτουργίας του σεναρίου και αφετέρου στην ικανότητα του αισθητήρα να λαμβάνει ορθές τιμές με την πάροδο του χρόνου.

Έχοντας ορίσει την θέση που πρέπει να φτάσει το μη επανδρωμένο εναέριο όχημα στα x=11m και y=11m η επιθυμητή κατάσταση είναι αυτή η οποία παρουσιάζεται στην Εικόνα 6.5-1.

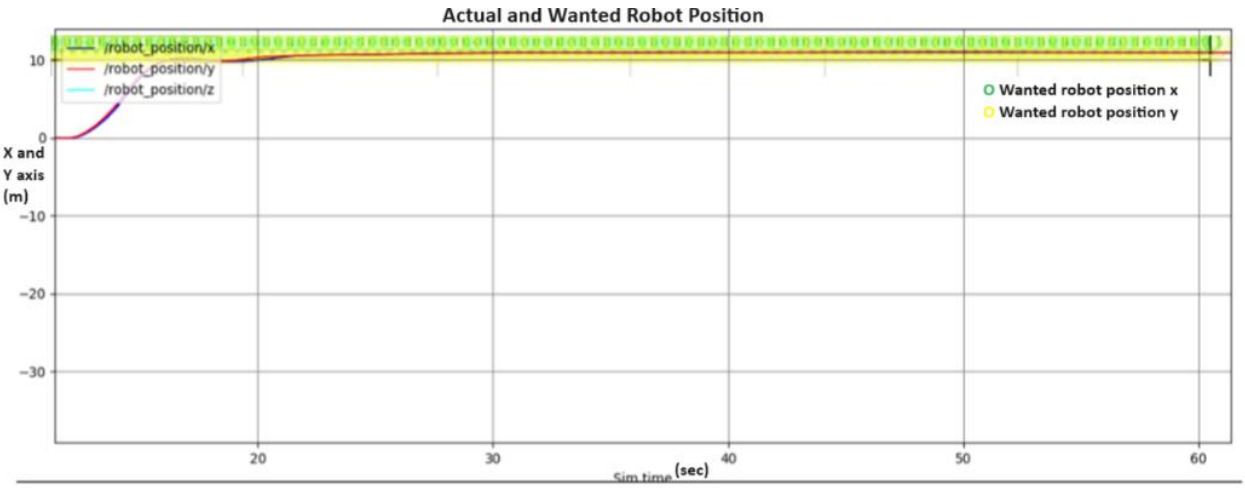



**Εικόνα 6.5-1: Επιθυμητή κατάσταση μέτρησης αισθητήρα UWB για σενάριο 5**

Την ίδια στιγμή, η κατάσταση η οποία λαμβάνεται από την προσομοίωση φαίνεται στην Εικόνα 6.5-2 και το ενοποιημένο τους διάγραμμα στην Εικόνα 6.5-3.



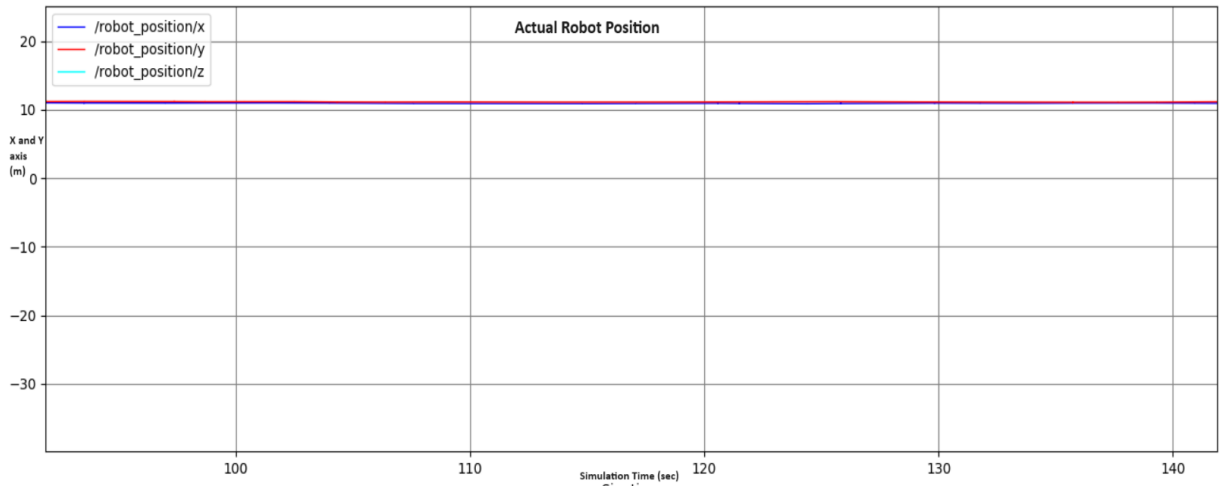

**Εικόνα 6.5-2: Μέτρηση αισθητήρα UWB κατά το σενάριο 5**



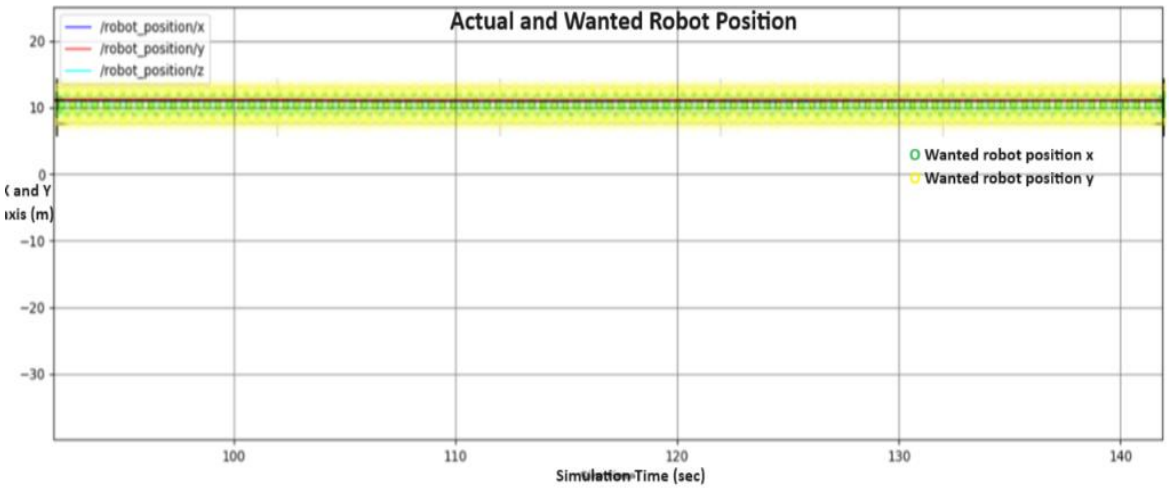

**Εικόνα 6.5-3: Ενοποιημένο διάγραμμα μετρήσεων αισθητήρα UWB για σενάριο 5**

Από αυτά, λαμβάνεται εύλογα το συμπέρασμα, ότι ο αισθητήρας UWB δεν παρουσιάζει αστάθειες κατά τον υπολογισμό της θέσης του drone παρόλο που αυτό έχει βρεθεί ελάχιστα έξω από τις καθορισμένες θέσεις. Επιπλέον, άξια παρατήρησης είναι η υπερακόντιση η οποία εμφανίζεται στα αρχικά δευτερόλεπτα της κίνησης του συστήματος και εικονίζεται στην Εικόνα 6.5-2. Αυτή καταδεικνύει την απογείωση του drone από την αρχική του θέση στο επιθυμητό ύψος.

Με σκοπό να ελεγχθεί εάν ο αισθητήρας θα μπορεί να υπολογίζει το ίδιο καλά την θέση του drone κατά την πάροδο του χρόνου λαμβάνεται το διάγραμμα που φαίνεται στην Εικόνα 6.5-4. Ταυτόχρονα, διατηρώντας σταθερή την επιθυμητή κατάσταση η οποία απεικονίζεται στην Εικόνα 6.5-1 προκύπτει και το ενοποιημένο διάγραμμα που είναι αυτό που παρουσιάζεται στην Εικόνα 6.5-5.



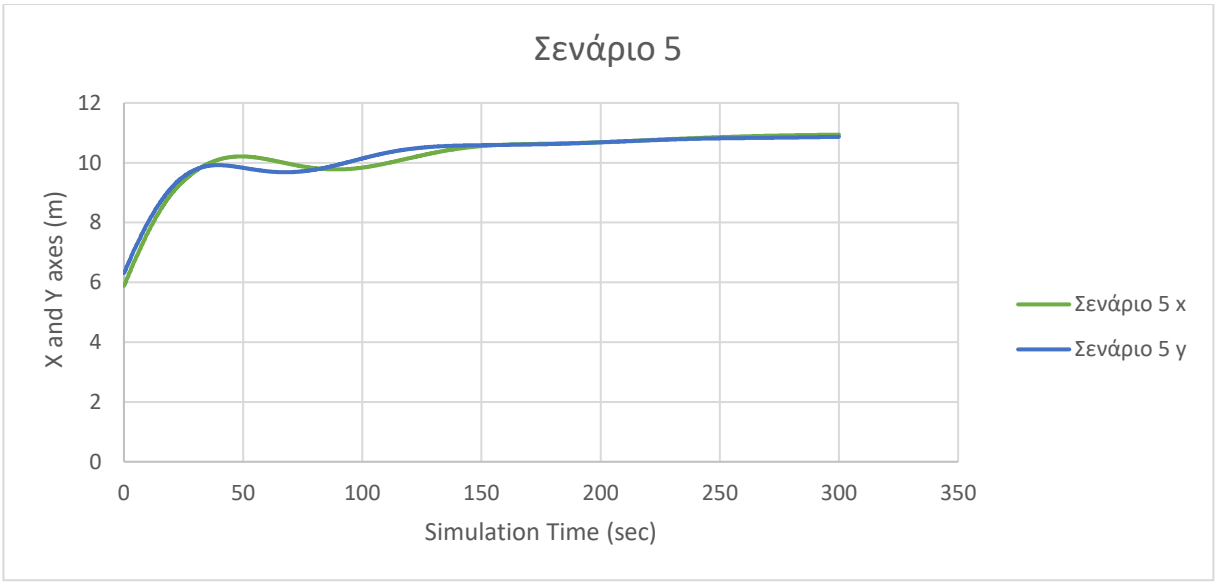

**Εικόνα 6.5-4: Μέτρηση αισθητήρα UWB κατά την πάροδο χρόνου στο σενάριο 5**




**Εικόνα 6.5-5: Ενοποιημένο διάγραμμα UWB κατά την πάροδο του χρόνου στο σενάριο 5**

Σύμφωνα με τα παραπάνω ο αισθητήρας UWB αντιλαμβάνεται την θέση του συστήματος με απόλυτη ακρίβεια και χωρίς διακυμάνσεις. Αυτό επιτυγχάνεται τόσο κατά την διάρκεια της εκκίνησης όσο και έπειτα από την παρέλευση ενός χρονικού διαστήματος.

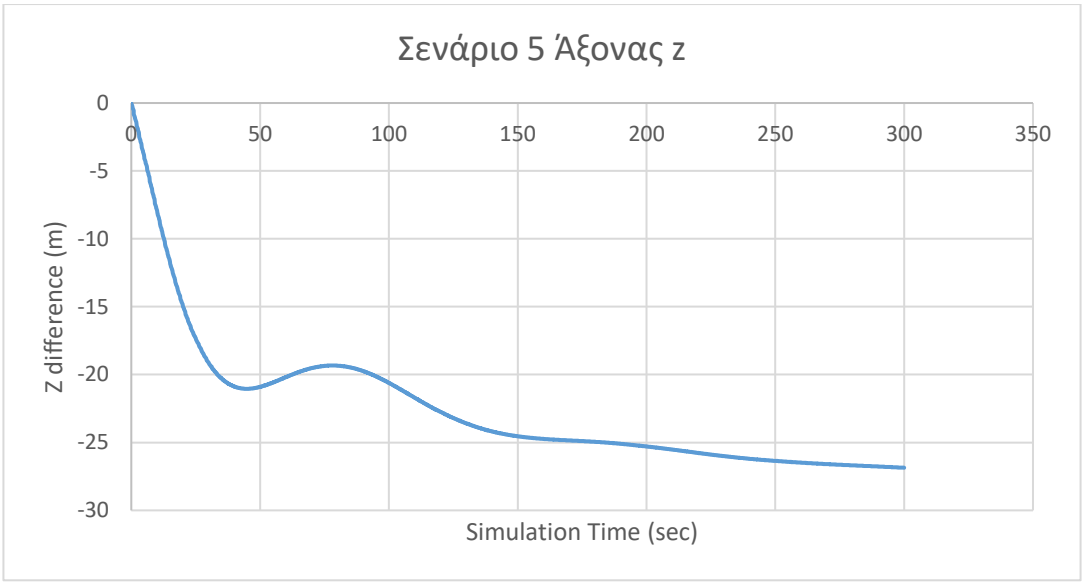
Για μεγαλύτερη επιβεβαίωση του παραπάνω συμπεράσματος είναι σημαντικό να δοθεί ένα διάγραμμα το οποίο θα απεικονίζει τη συνολική κίνηση του drone. Αυτό απεικονίζεται στην Εικόνα 6.5-6.




**Εικόνα 6.5-6:Συνολική κίνηση του drone κατά το σενάριο 5**

Από το παραπάνω διάγραμμα συμπεραίνεται ότι οι μετρήσεις του αισθητήρα εκτός των ορίων των αγκυρών είναι αρκετά ακριβείς. Πιο αναλυτικά, κατά την αρχή της κίνησης παρατηρούνται αστάθειες για την εύρεση της θέσης στους άξονες x και y από τον προσομοιωμένο αισθητήρα UWB. Μετά την παρέλευση χρόνου, ωστόσο, οι τιμές φαίνεται να σταθεροποιούνται στις επιθυμητές χωρίς την ύπαρξη διακυμάνσεων.

Σημαντικό θεωρείται να δειχθούν οι μετρήσεις του αισθητήρα για τον τρίτο άξονα, αυτόν του z’z. Οι υπολογισμοί σε αυτό το επίπεδο δεν είναι λογικοί και απέχουν από την προκαθορισμένη τιμή των 2m. Εξαιτίας αυτού, παρέχεται το διάγραμμα με τις διαφορές μεταξύ των μετρήσεων με σκοπό να γίνει κατανοητή η αντίληψη που διαθέτει ο αισθητήρας. Αυτό φαίνεται στην Εικόνα 6.5-7.



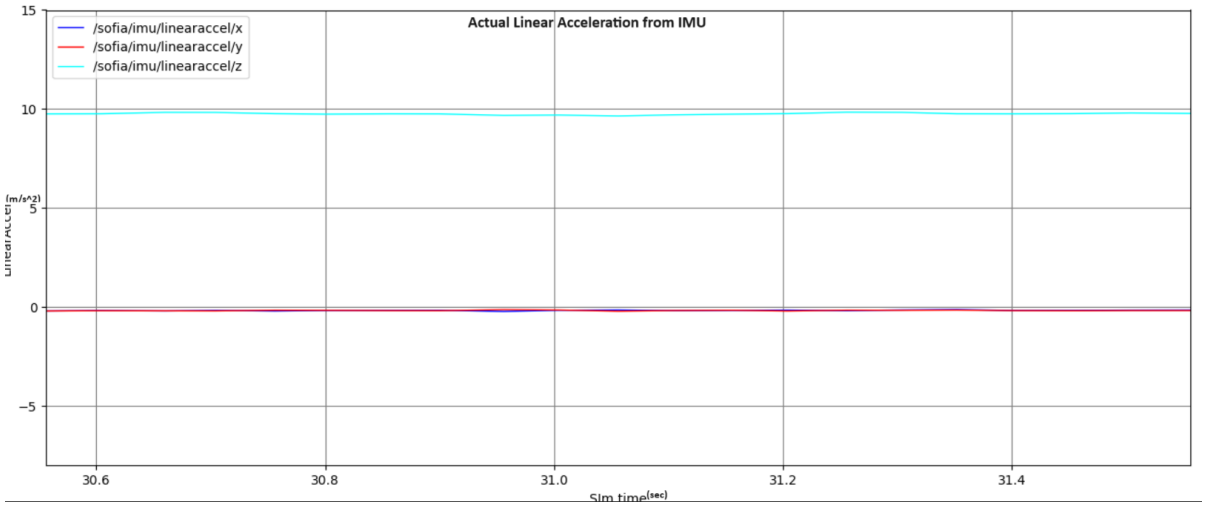

**Εικόνα 6.5-7: Διαφορές των τιμών του άξονα z στο σενάριο 5**

Από το παραπάνω διάγραμμα παρατηρείται ότι η μέτρηση του ύψους δεν διατηρείται σταθερή αλλά συνεχώς αυξάνεται. Το γεγονός αυτό σημαίνει ότι ο υπολογισμός του ύψους δεν συγκλίνει σε μία τιμή αλλά μεταβάλλεται. Αυτό δεν είναι ορθό καθώς στο συγκεκριμένο σενάριο το ύψους διατηρείται σταθερό σε μια προκαθορισμένη τιμή. Επομένως, ο υπολογισμός των τιμών του άξονα z είναι αναξιόπιστες.

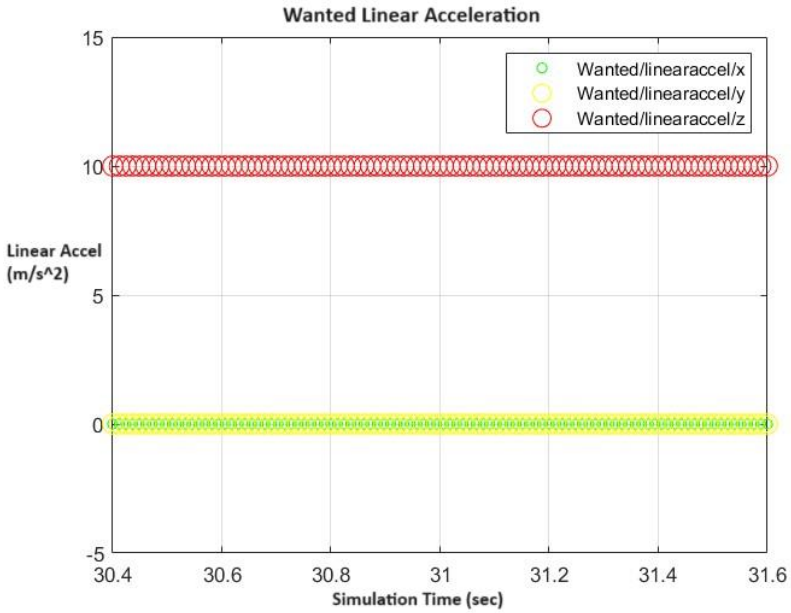
### 6.5.2 Ο αισθητήρας IMU στο σενάριο 5

Με στόχο την ολοκληρωμένη μελέτη του πέμπου σεναρίου οφείλονται να συζητηθούν οι τιμές της γραμμικής επιτάχυνσης και των ταχυτήτων οι οποίες υπολογίζονται από τον ενσωματωμένο αισθητήρα IMU.

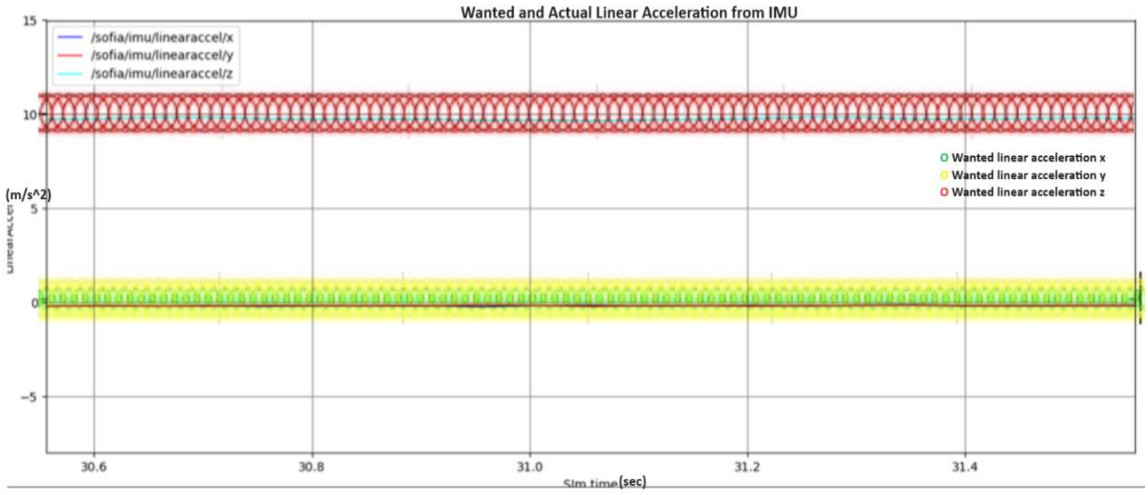
Πιο συγκεκριμένα, ξεκινώντας την αναφορά από την μέτρηση της γραμμικής επιτάχυνσης αυτή φαίνεται στην Εικόνα 6.5-8. Η μέτρηση αυτή θα πρέπει να συμβαδίζει με τις προσομοιωμένες τιμές και η επιθυμητή κατάστασή της παρουσιάζεται στην Εικόνα 6.5-9. Παράλληλα, τοποθετείται και το ενοποιημένο διάγραμμα στην Εικόνα 6.5-10 με σκοπό την καλύτερη κατανόηση. Οι μετρήσεις αυτές πρέπει να είναι σταθερές ανεξαρτήτως χρονικού διαστήματος καθώς έπειτα από την απογείωση του το drone διατηρεί την θέση του σταθερή.




**Εικόνα 6.5-8: Μέτρηση γραμμικής επιτάχυνσης στο σενάριο 5**




**Εικόνα 6.5-9: Επιθυμητή κατάσταση γραμμικής επιτάχυνσης στο σενάριο 5**

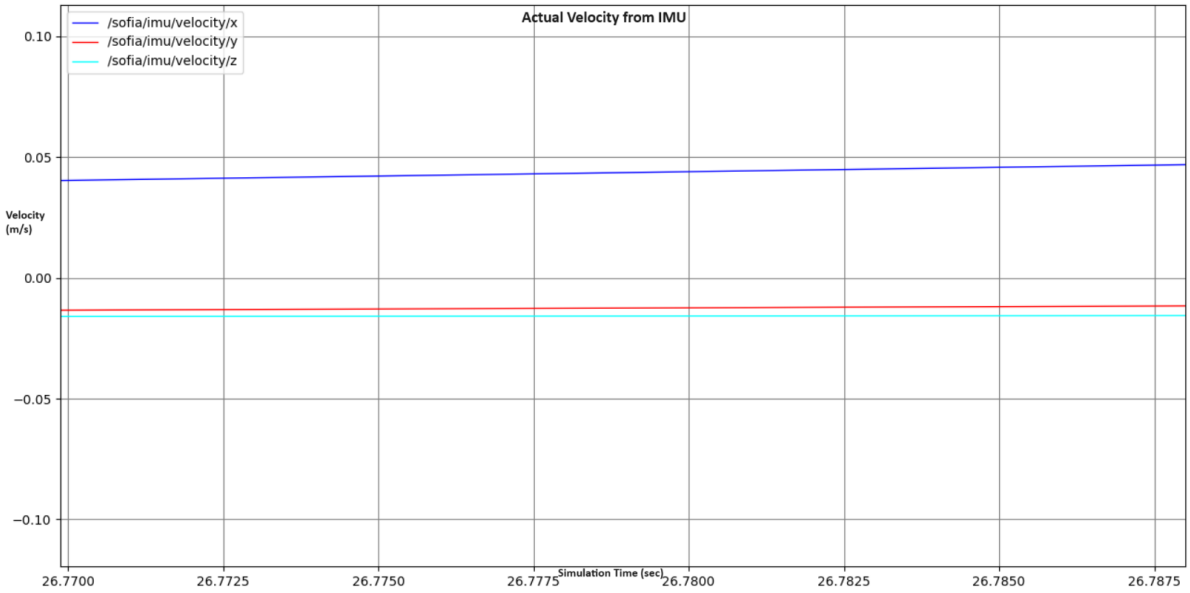



**Εικόνα 6.5-10: Ενοποιημένο διάγραμμα γραμμικής επιτάχυνσης στο σενάριο 5**

Οι προσομοιωμένες μετρήσεις αντιπροσωπεύουν την πραγματικότητα και βρίσκονται πολύ κοντά στις καθορισμένες πραγματικές τιμές. Παρατηρούνται, ωστόσο, μικρές αποκλίσεις των τιμών προς τα κάτω χωρίς να θεωρούνται αξιοσημείωτες.

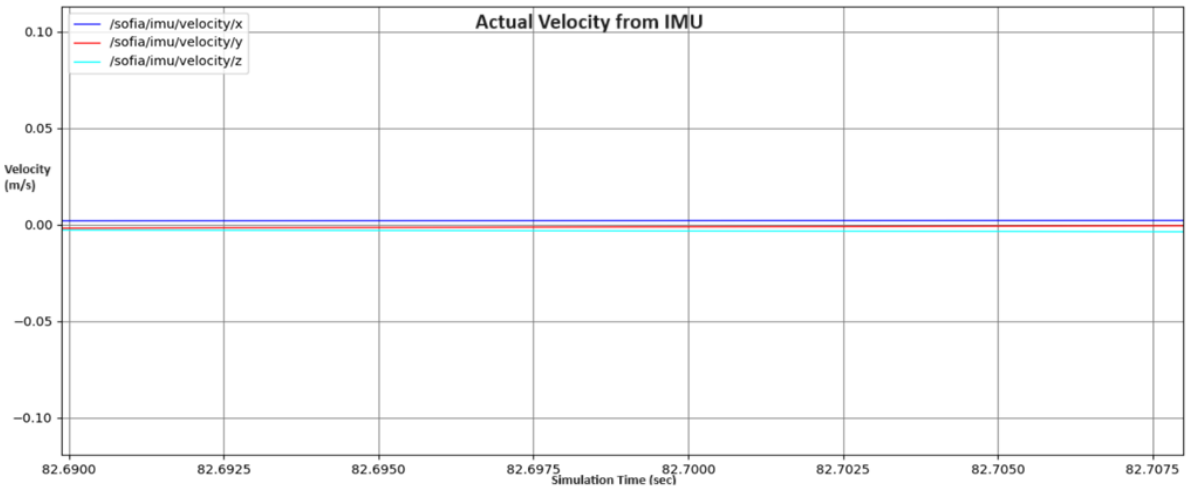
Συνεχίζοντας την μελέτη με την μέτρηση των ταχυτήτων εξετάζεται αφενός το ενδεχόμενο παρακολούθησης της κίνησης και αφετέρου η ικανότητα του αισθητήρα να μετράει σωστά και σταθερά τις απαιτούμενες τιμές.

Αναλυτικότερα, στην Εικόνα 6.5-11 παρουσιάζεται ένα στιγμιότυπο κατά το οποίο το μη επανδρωμένο εναέριο όχημα ξεκινά την επιτάχυνσή του έτσι ώστε να φτάσει στην τελική του θέση. Το γεγονός αυτό αποδεικνύει την ικανότητα του αισθητήρα να παρακολουθεί με ορθό τρόπο όλες τις μεταβολές στην κίνηση του αεροσκάφους.

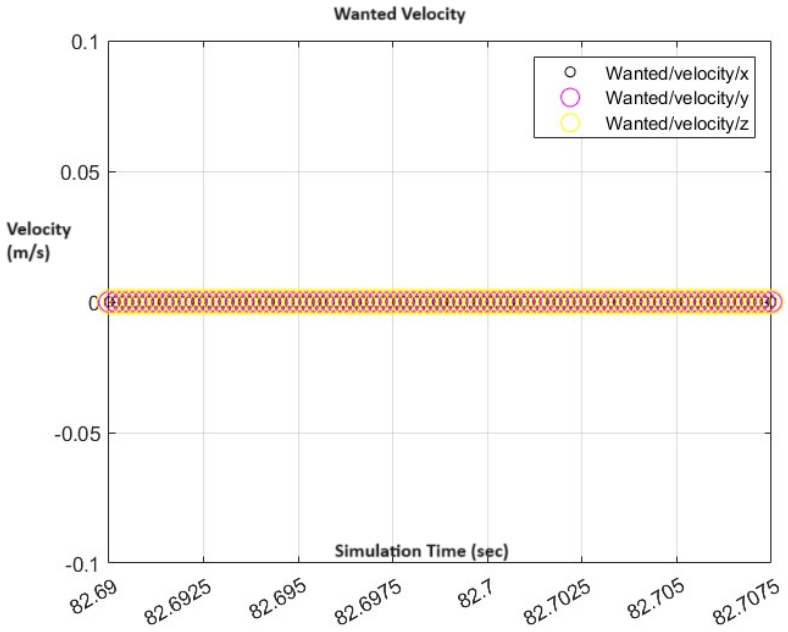



**Εικόνα 6.5-11: Διάγραμμα ταχυτήτων κατά την εκκίνηση στο σενάριο 5**

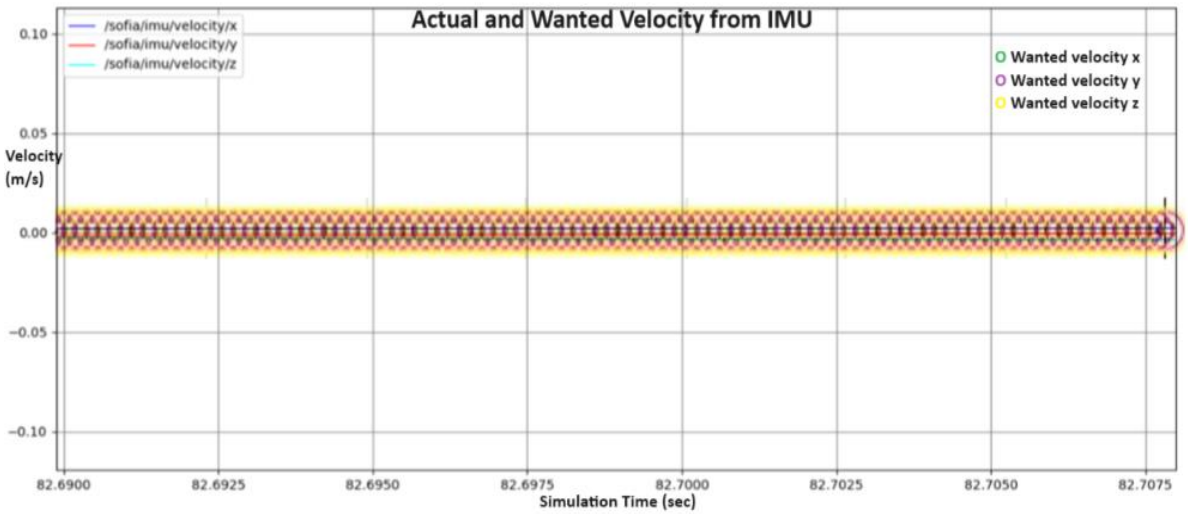
Ωστόσο, για τον έλεγχο της σταθερότητας και της ακρίβειας των μετρήσεων θεωρείται πρέπον να χρησιμοποιηθεί ένα χρονικό διάστημα κατά το οποίο το drone θα βρίσκεται σίγουρα ακίνητο σε σταθερή θέση. Επομένως, το προσομοιωμένο αποτέλεσμα φαίνεται στην Εικόνα 6.5-12. Ταυτόχρονα, στην Εικόνα 6.5-13 φαίνεται η επιθυμητή κατάσταση και στην Εικόνα 6.5-14 το ενοποιημένο διάγραμμα.




**Εικόνα 6.5-12: Διάγραμμα ταχυτήτων σε σταθερή θέση στο σενάριο 5**




**Εικόνα 6.5-13: Επιθυμητή κατάσταση ταχυτήτων σε σταθερό σημείο στο σενάριο 5**




**Εικόνα 6.5-14: Ενοποιημένο διάγραμμα ταχυτήτων σε σταθερό σημείο στο σενάριο 5**

Ο αισθητήρας καταλαβαίνει την κατάσταση των ταχυτήτων του drone με απόλυτη ακρίβεια. Το συμπέρασμα αυτό εξάγεται καθώς δεν υπάρχουν αποκλίσεις κατά την μέτρηση σε σταθερή θέση. Παράλληλα, είναι ικανός να αντιλαμβάνεται πλήρως τις αλλαγές στην κινητική κατάσταση του συστήματος.

## 6.6 Ο Προσανατολισμός του drone

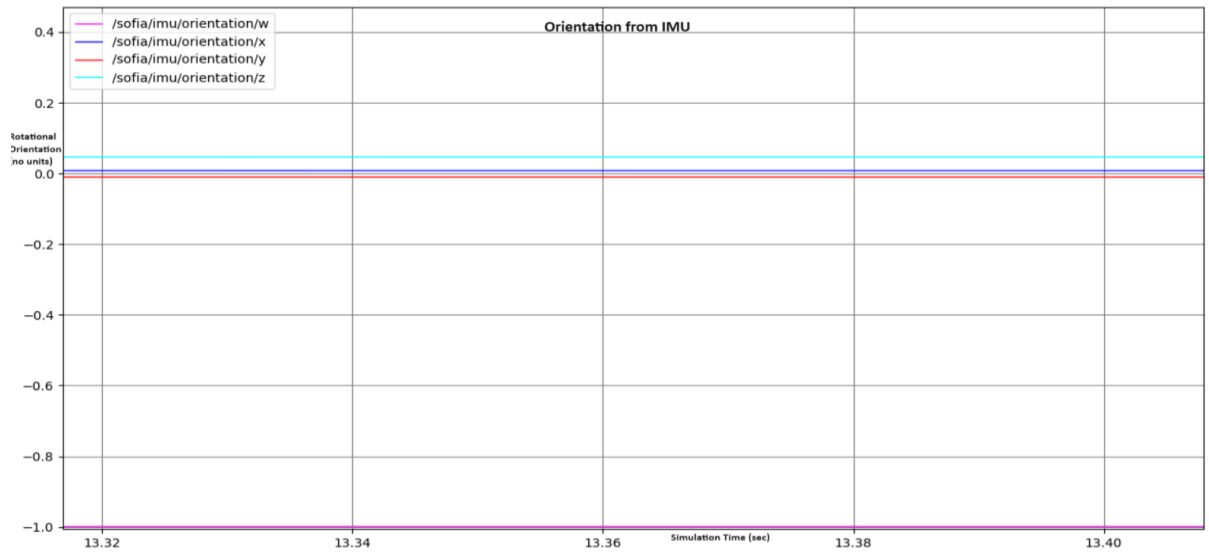
Με στόχο την ολοκληρωμένη μελέτη του αισθητήρα IMU θα πρέπει να αποδοθούν διαγράμματα τα οποία θα αναφέρονται στον προσανατολισμό του drone καθώς είναι ένα μέγεθος το οποίο υπολογίζεται με τον εν λόγω αισθητήρα.

Ο προσανατολισμός του drone είναι ανεξάρτητος του σεναρίου που εκτελείται. Αυτό γίνεται καθώς, καθορίζεται από συγκεκριμένη εντολή στον κώδικα και ρυθμίζεται από τον χρήστη. Στο παρόν κεφάλαιο δίνεται ενδεικτικά το παράδειγμα δύο σεναρίων, του σεναρίου 1 και του σεναρίου 3.

### 6.6.1 Ο προσανατολισμός του drone στο σενάριο 1

Σκοπός της μελέτης είναι ο έλεγχος αντίληψης του αισθητήρα στην μεταβολή του προσανατολισμού του μη επανδρωμένου εναέριου οχήματος. Πιο αναλυτικά, στην Εικόνα 6.6-1 παρουσιάζεται ο προσανατολισμός του

drone πριν την εκκίνησή του όσο αυτό βρίσκεται στην προκαθορισμένη θέση του. Η θέση αυτή είναι ορισμένη από τον αρχικό κώδικα της εφαρμογής.




**Εικόνα 6.6-1: Προκαθορισμένος προσανατολισμός του drone στο σενάριο 1**

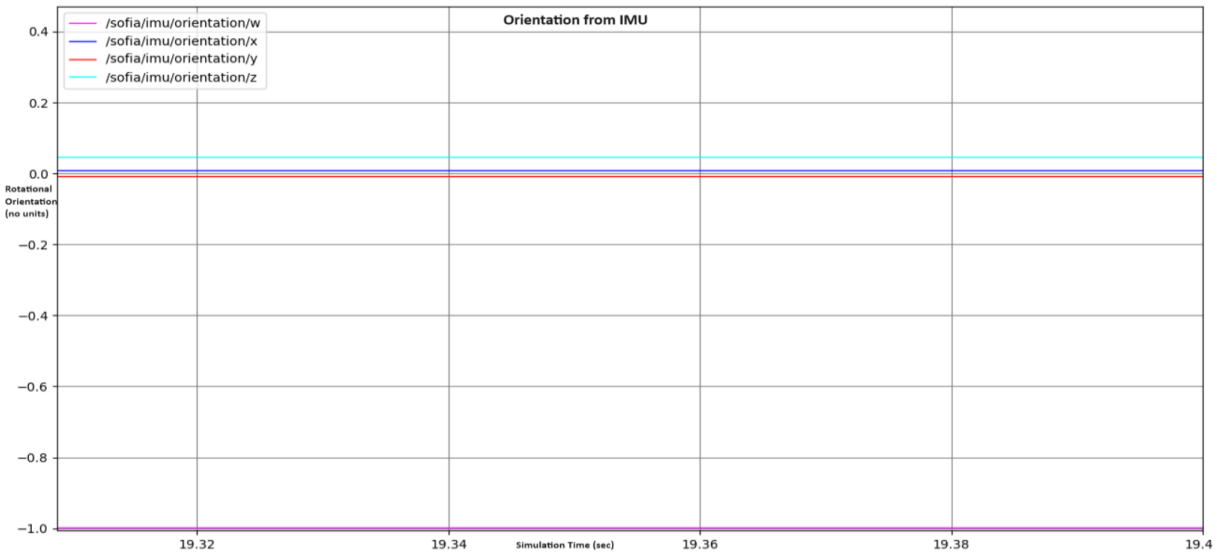
Δίνοντας εντολή για μια τυχαία αλλαγή του προσανατολισμού του drone το σύστημα φαίνεται να ανταποκρίνεται θετικά. Αναλυτικότερα, το μη επανδρωμένο εναέριο όχημα πραγματοποιεί μία κίνηση στρέψης και έπειτα από ελάχιστα δευτερόλεπτα λαμβάνει την κατεύθυνση που φαίνεται στην Εικόνα 6.6-2. Με αυτόν τον τρόπο ο αισθητήρας IMU έχει την ικανότητα να παρακολουθεί με ακρίβεια και την αλλαγή του προσανατολισμού.


**Εικόνα 6.6-2: Προσανατολισμός του drone έπειτα από σταθεροποίησή του στην τελική θέση του σεναρίου 1**

### 6.6.2 Ο προσανατολισμός του drone κατά το σενάριο 3

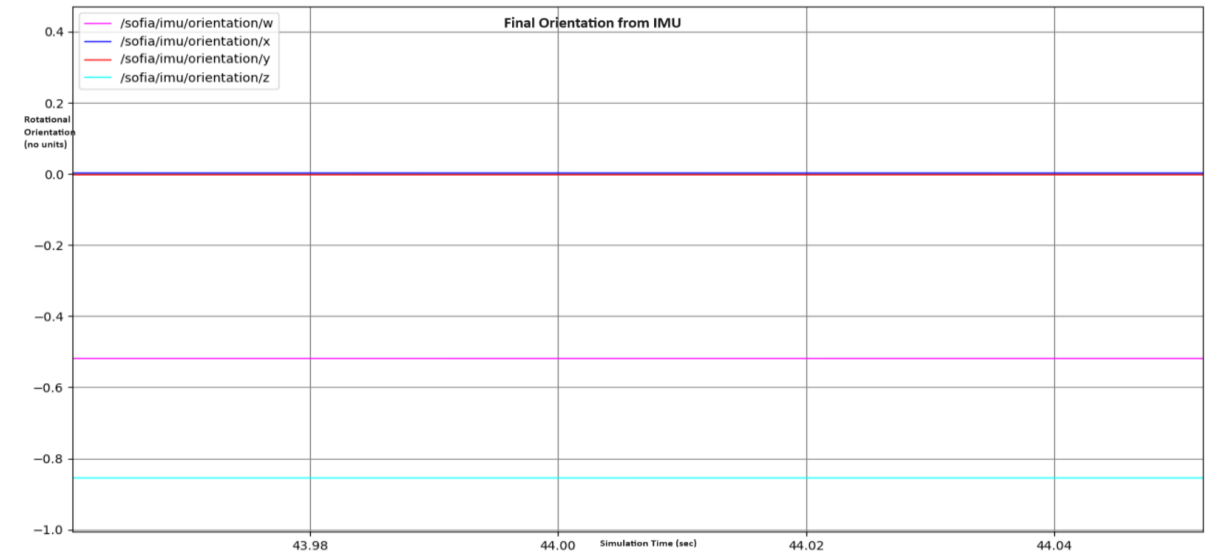
Το παρόν κεφάλαιο έχει σκοπό να αποδείξει ότι η μέτρηση του προσανατολισμού του ρομποτικού συστήματος είναι ανεξάρτητη από το σενάριο προσομοίωσης.

Πιο αναλυτικά, ξεκινώντας την αναφορά ο προσανατολισμός του drone πριν την εκκίνησή του όσο αυτό βρίσκεται στην προκαθορισμένη θέση του φαίνεται στην Εικόνα 6.6-3. Αυτός είναι ακριβώς ίδιος με του σεναρίου 1 καθώς οι εντολές αρχικής θέσης παραμένουν αμετάβλητες.




**Εικόνα 6.6-3: Προκαθορισμένος προσανατολισμός του drone στο σενάριο 3**

Αναθέτοντας τις ίδιες εντολές κατά την εκτέλεση του τρίτου σεναρίου το αποτέλεσμα του προσανατολισμού έπειτα από την παρέλευση ενός χρονικού διαστήματος φαίνεται στην Εικόνα 6.6-4. Ο προσανατολισμός που έλαβε το drone στο τέλος της κίνησής του είναι ακριβώς ο ίδιος με το προηγούμενο σενάριο. Αυτό καταδεικνύει έντονα πως η μέτρηση του μεγέθους του προσανατολισμού καθορίζεται από τις εντολές που δίνονται και δεν εξαρτάται από τα σενάρια προσομοίωσης.




**Εικόνα 6.6-4: Προσανατολισμός του drone στην τελική θέση του σεναρίου 3**

# 7 ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

## 7.1 Συμπεράσματα

Η παρούσα διπλωματική εργασία είχε διττό χαρακτήρα. Αναλυτικότερα, στα πλαίσια αυτής εξετάστηκαν αφενός ο σχεδιασμός και η κατασκευή των ποδιών του μη επανδρωμένου εναέριου οχήματος και αφετέρου η ένταξη αισθητήρων και η λειτουργία τους στην πλοήγηση του drone σε εσωτερικό χώρο.

### 7.1.1 Συμπεράσματα για τα μηχανικά μέρη

Ξεκινώντας με το πρώτο μέρος,τα πόδια τα οποία κατασκευάσθηκαν στα πλαίσια της εκτέλεσης της διπλωματικής εργασίας άλλαξαν φιλοσοφία και λογική. Πιο συγκεκριμένα, οι αλλαγές αφορούσαν το υλικό κατασκευής, την απαιτούμενη κατεργασία, το σχήμα και τη μέθοδο σύνδεσης με το κύριο σώμα του drone. Αναφέροντας συνοπτικά, το υλικό άλλαξε από αλουμίνιο σε PLA, η κατεργασία από φρέζα σε τεχνική τρισδιάστατης εκτύπωσης, το σχήμα από τραπεζοειδές σε σφαιρικό, και η σύνδεση από κοχλιωτή σε εφαρμοστή με ένωση κόλλας.

Η αλλαγή του υλικού από αλουμίνιο σε PLA έχει σημαντικές επιπτώσεις στο βάρος του drone. Το PLA είναι ένα πλαστικό που χαρακτηρίζεται από χαμηλότερη πυκνότητα και βάρος σε σύγκριση με το αλουμίνιο. Αυτό είχε ως αποτέλεσμα τη μείωση του συνολικού βάρους του drone, που δύναται να επιφέρει βελτίωση στην αυτονομία πτήσης και στην ευελιξία του εναέριου οχήματος.

Επιπλέον, αξιοσημείωτη είναι και η μεταβολή της παραδοσιακής κατεργασίας της φρέζας με την τεχνική της τρισδιάστατης εκτύπωσης. Αυτή έχει να επιφέρει ως βασικό αποτέλεσμα την αύξηση της ταχύτητας παραγωγής και τη μείωση του κόστους. Η τρισδιάστατη εκτύπωση επιτρέπει την ταχύτερη κατασκευή εξαρτημάτων με μεγαλύτερη ακρίβεια. Επίσης, η τεχνική αυτή έχει προσφέρει την ευκολία κατασκευής καθώς δεν απαιτείται η ανάμειξη με σύνθετα μηχανήματα και πολύπλοκα εργαλεία.

Ταυτόχρονα, η αλλαγή του σχήματος των ποδιών από τραπεζοειδές σε σφαιρικό δύναται να προσφέρει μια διαφορετική οπτική στο σχεδιασμό. Αναλυτικότερα, το νέο σχήμα δημιουργήθηκε με την λογική της βελτιστοποίησης της υπάρχουσας φιλοσοφίας του drone. Ο σχεδιασμός τους είναι θεωρητικός και η κατασκευή τους δεν πραγματοποιήθηκε στα πλαίσια της διπλωματικής εργασίας. Παρόλα αυτά, το νέο τους σφαιρικό σχήμα έχει προοπτικές να συντελέσει στη διόρθωση ζητημάτων ισορροπίας του συστήματος. Αυτό μπορεί να συμβεί καθώς οι επιφάνειες είναι περισσότερο ευέλικτες και ικανές να παραλάβουν τις αστάθειες της κίνησης.

Ολοκληρώνοντας την αναφορά στις μεταβολές των ποδιών είναι σημαντικό να αναλυθεί και η μέθοδος σύνδεσης όλων των εξαρτημάτων τους. Αυτή αλλάζει από σύνδεση με κοχλίες σε εφαρμοστή εισχώρηση και στερεοποίηση με ειδική κόλλα. Η σύνδεση αυτή υποβλήθηκε στο σύστημα για την αντιμετώπιση πρακτικών προβλημάτων. Δηλαδή, οι καινούργιοι δοκοί που είναι ενισχυμένοι με ίνες άνθρακα δεν έχουν την ικανότητα διάνοιξης οπών. Με αυτόν τον τρόπο, η μόνη δυνατή λύση είναι η προσαρμογή των εξαρτημάτων με κόλληση. Ένα βασικό μειονέκτημα αυτής της τεχνικής είναι ο κίνδυνος εξασθένισης της δύναμης της κόλλας σε ορισμένο χρονικό διάστημα λόγω των συνθηκών του περιβάλλοντος όπως η υγρασία.

### 7.1.2 Συμπεράσματα του κώδικα και των προσομοιώσεων

Κατά την εκτέλεση του δεύτερου μέρους της διπλωματικής εργασίας σημειώνεται ότι ζητούμενο ήταν η ενσωμάτωση στο drone δύο βασικών αισθητήρων αυτών του γυροσκοπίου (IMU) και του ultra wide band (UWB) αλλά και η δοκιμή αυτών μέσω προσομοιώσεων. Ο κώδικας για τον αισθητήρα UWB έχει γραφτεί σε C++, ενώ το IMU είναι μια προσφερόμενη προσθήκη από το λογισμικό Gazebo. Οι προσομοιώσεις που έχουν πραγματοποιηθεί αποσκοπούν στην αξιολόγηση της αποτελεσματικότητας των αισθητήρων σε διάφορα σενάρια πτήσης.

Ο κώδικας ο οποίος αναφέρεται στον αισθητήρα UWB φαίνεται να είναι εξαιρετικά αποτελεσματικός στην παρακολούθηση της πορείας του drone. Αυτό συμβαίνει καθώς σε όλες τις προσομοιώσεις ακολουθεί την πορεία του drone με ακρίβεια, ανεξαρτήτως της πολυπλοκότητας των κινήσεων. Ακόμα και σε σενάρια με σύνθετες πορείες, όπου αυτό πραγματοποιούσε απότομες στροφές και αλλαγές κατεύθυνσης, ο αισθητήρας δεν έχασε το σήμα του οχήματος.

Παρόλα αυτά, παρατηρήθηκαν διακυμάνσεις στα διαγράμματα που προέκυψαν από τα δεδομένα του αισθητήρα UWB. Οι διακυμάνσεις αυτές μπορεί να αποδοθούν σε παρεμβολές σήματος. Ωστόσο, δεν επηρέασαν την συνολική απόδοση της παρακολούθησης, καθώς το σήμα παρέμεινε επαρκές για την αξιόπιστη παρακολούθηση του drone.

Επιπλέον, ο αισθητήρας IMU αντιλαμβάνεται και αυτός με ακρίβεια τα μεγέθη της γραμμικής επιτάχυνσης, της ταχύτητας και του προσανατολισμού του drone. Αναλυτικότερα, οι προσομοιώσεις έδειξαν ότι αυτό διαβάζει και υπολογίζει με σωστό τρόπο και σε όλη την διάρκεια της κίνησης τα σήματα που σχετίζονται με τα προαναφερόμενα μεγέθη. Την ίδια στιγμή, εμφανίζει ομαλότερες διακυμάνσεις στα διαγράμματά του από αυτές του προηγούμενου αισθητήρα UWB γεγονός που υποδηλώνει σταθερότητα στις μετρήσεις. Παράλληλα, το IMU παρακολούθησε σωστά και τον προσανατολισμό του οχήματος. Η καταγραφή των γωνιακών μεταβολών και της κατεύθυνσης ήταν ακριβής, συμβάλλοντας στην καλύτερη κατανόηση της θέσης και της κίνησης του drone στον τρισδιάστατο χώρο.

Ωστόσο, το IMU αντιμετώπισε δυσκολίες κατά την παρακολούθηση πιο σύνθετων κινήσεων του drone. Πιο συγκεκριμένα, σε σενάρια όπου το εναέριο όχημα εκτελούσε περίπλοκες κινήσεις, το IMU δεν κατάφερε να καταγράψει πλήρως τα μεγέθη της κίνησης. Η ακρίβεια των μετρήσεων του IMU ήταν υψηλή όταν το drone ήταν σε σταθερή πτήση, σε στάση ή σε απλή μετάβαση, αλλά παρουσίασε ελλείψεις κατά την καταγραφή δυναμικών κινήσεων.

Το πρόβλημα αυτό, ωστόσο, δύναται να διορθωθεί. Πιο αναλυτικά, στον κώδικα που έχει γραφεί οι μετρήσεις λαμβάνονται με συχνότητα η οποία είναι ίση με 10Hz. Αυτή κρίνεται ανεπαρκής στις δυναμικές κινήσεις καθώς το σύστημα δεν είναι ικανό να αντιληφθεί τις μεταβολές σε τόσο λίγο χρονικό διάστημα. Εξαιτίας αυτού, θεωρήθηκε σωστό να μεταβληθεί αυτή η συχνότητα καταγραφής. Ειδικότερα, η μεταβολή ξεκίνησε συντηρητικά με αλλαγή των 10 Hz σε 100 Hz. Λειτουργώντας το σύστημα αρκετές φορές, παρατηρήθηκε ότι ούτε η νέα συχνότητα είχε την δυνατότητα να παρακολουθεί πλήρως τις μεταβολές της κίνησης παρόλο που φάνηκε να παρουσιάζονται βελτιώσεις στις καταγραφές. Αυτό οδήγησε σε νέα αλλαγή της τιμής της. Αυτή ήταν η αύξηση από τα 100 Hz στα 1000Hz. Με αυτή την νέα συχνότητα ο αισθητήρας IMU έχει την δυνατότητα να παρακολουθεί και την παραμικρή μεταβολή στις κινήσεις του συστήματος τόσο στις δυναμικές όσο και στις στατικές. Αυτό αυξάνει κατακόρυφα την ακρίβεια ολόκληρης της ρομποτικής εφαρμογής.

Η συνδυαστική χρήση των αισθητήρων UWB και IMU παρέχει μια ολοκληρωμένη εικόνα της κίνησης και της θέσης του drone. Ο αισθητήρας UWB διακρίνεται για την ικανότητά του να παρακολουθεί την πορεία του drone με ακρίβεια, ακόμα και σε σύνθετα σενάρια, ενώ το IMU προσφέρει ομαλές και σταθερές μετρήσεις γραμμικής επιτάχυνσης, ταχύτητας και προσανατολισμού. Το γεγονός αυτό καλύπτει όλες τις περιπτώσεις κίνησης του drone στον τρισδιάστατο χώρο. Την ίδια στιγμή, τα αποτελέσματα δείχνουν ότι η ενσωμάτωση και των δύο αισθητήρων είναι αυτή που είναι ικανή να δημιουργήσει την βέλτιστη απόδοση του εναέριου οχήματος. Αυτό γίνεται καθώς ο ένας αισθητήρας καλείται να καλύψει τα κενά του άλλου τόσο σε ζητήματα κινήσεων όσο και σε θέματα ακρίβειας.

Ολοκληρώνοντας, αξίζει να σημειωθεί η δυσκολία δημιουργίας της επιφάνειας εργασίας για έναν νέο προγραμματιστή. Πιο αναλυτικά, το λογισμικό που χρησιμοποιήθηκε κατά την διπλωματική εργασία αξιοποιεί το λειτουργικό σύστημα των Linux και συγκεκριμένα τα Ubuntu 20.04. Αυτό σημαίνει ότι, αρχικά, το νέο αυτό σύστημα θα πρέπει να εγκατασταθεί στον προσωπικό υπολογιστή και να ρυθμιστεί έτσι ώστε να λειτουργεί παράλληλα με αυτό των Windows. Ταυτόχρονα, θα πρέπει να εγκατασταθούν στον ίδιο υπολογιστή οι έτοιμοι κώδικες τους οποίους προσφέρει η εταιρία PX4. Αυτοί διαθέτουν όλες τις πληροφορίες

που χρειάζονται σχετικά με τον ελεγχτή της πτήσης, το drone iris και τις προσθήκες όπως αυτή του γυροσκοπίου. Όλα αυτά βρίσκονται σε μια καθορισμένη επιφάνεια εργασίας η οποία λειτουργεί αυτόματα έπειτα από την εγκατάστασή της.

Έχοντας υλοποιήσει τα παραπάνω, πρέπει να δημιουργηθεί η προσωπική επιφάνεια εργασίας του χρήστη. Συγκεκριμένα, χρειάζεται να ανοιχθεί ένας νέος φάκελος στο περιβάλλον των Linux με προτεινόμενο όνομα «catkin_workspace». Αυτός θα οργανωθεί με την βοήθεια υπό φακέλων οι οποίοι έχουν ως στόχο την αποθήκευση στο εσωτερικό τους των βασικών αρχείων για την ορθή λειτουργία του προγράμματος. Πιο αναλυτικά, αυτοί είναι οι «.catkin_tools», «build»., «devel», «logs» και «src». Ως πιο σημαντικός από αυτούς είναι ο τελευταίος καθώς στο εσωτερικό του είναι αποθηκευμένα τόσο ο κόμβος ο οποίος γράφεται όσο και τα συμπληρωματικά αρχεία κώδικα τα οποία τον βοηθούν να λειτουργήσει με αποτελεσματικότητα. Τέλος, οι υπόλοιποι υπό φάκελοι διαθέτουν αρχεία απαραίτητα για την πλαισίωση του συστήματος τα οποία όμως δεν χρησιμοποιούνται ενεργά από τον εκάστοτε χρήστη.

## 7.2 Μελλοντική εργασία

Η παρούσα διπλωματική εργασία επικεντρώνεται στη μελέτη και την ανάπτυξη τεχνικών πλοήγησης ενός drone σε εσωτερικό χώρο, χρησιμοποιώντας δύο βασικούς αισθητήρες: τον αισθητήρα Ultra wide band (UWB) και το γυροσκόπιο (IMU). Η εφαρμογή αυτών των τεχνολογιών αποσκοπεί στην επίτευξη ακριβούς και αξιόπιστης πλοήγησης σε περιβάλλοντα όπου η χρήση του GPS είναι ανεπαρκής ή αδύνατη. Μελλοντικές επεκτάσεις της εργασίας αυτής μπορούν να βελτιώσουν περαιτέρω την απόδοση και τη λειτουργικότητα του συστήματος πλοήγησης αλλά και να το δοκιμάσουν σε αληθινές συνθήκες. Αυτές περιλαμβάνουν την ακριβή προσγείωση και απογείωση, την ανάπτυξη κώδικα για την αντιμετώπιση εμποδίων, την ενσωμάτωση επιπλέον φίλτρων, και την πειραματική διάταξη για πτήσεις σε πραγματικές συνθήκες.

Ξεκινώντας με την πρώτη μελλοντική επέκταση θα αναλυθεί η ακριβής προσγείωση και απογείωση του drone σε επίπεδο προσομοίωσης. Αναλυτικότερα, οι δύο αυτές διεργασίες είναι κρίσιμες φάσεις της πτήσης του drone καθώς οποιαδήποτε ανακρίβεια μπορεί να οδηγήσει σε απώλεια ισορροπίας με πιθανό το σενάριο βλάβης. Εξαιτίας αυτού στα πλαίσια της περαιτέρω ανάπτυξης της εργασίας δύναται να αναπτυχθούν αλγόριθμοι που θα επιτρέπουν τον ακριβή έλεγχο της θέσης και της ταχύτητας του drone κατά την προσέγγιση και την απομάκρυνση από την βάση του. Αυτό μπορεί να πραγματοποιηθεί με την ενσωμάτωση επιπλέον αισθητήρων όπως υπερηχητικών αισθητήρων ή και μίας απλής κάμερας οι οποίοι έχουν την δυνατότητα να βελτιώσουν την ακρίβεια του συστήματος κατά τις φάσεις αυτές.

Στη συνέχεια, το ενδιαφέρον στρέφεται στην δεύτερη κατά σειρά μελλοντική επέκταση η οποία αφορά την ανάπτυξη του κώδικα έτσι ώστε να δοθεί η δυνατότητα στο drone να μπορεί να ανιχνεύει και να αποφεύγει εμπόδια στον εσωτερικό χώρο. Η ικανότητα αυτή είναι ζωτικής σημασίας για την ασφαλή πλοήγηση σε βιομηχανικούς χώρους όπου τα μηχανήματα είναι πολλά και τοποθετημένα σε τυχαίες θέσεις. Για να γίνει αυτό θα πρέπει ο ήδη υπάρχον κώδικας να ενισχυθεί με περαιτέρω ανάπτυξη αλγορίθμων οι οποίοι θα επιτρέπουν στο μη επανδρωμένο εναέριο όχημα να αναγνωρίζει τη θέση των εμποδίων και να εντοπίζει τις βέλτιστες διαδρομές για την αποφυγή τους.

Παράλληλα, εξαιρετικό ενδιαφέρον θα έχει και η περεταίρω ανάπτυξη του κώδικα με την ενσωμάτωση περισσότερων φίλτρων όπως το φίλτρο Κάλμαν. Αυτό είναι μια αλγοριθμική προσέγγιση που χρησιμοποιείται για την εκτίμηση της κατάστασης ενός συστήματος σε πραγματικό χρόνο, λαμβάνοντας υπόψη τις μετρήσεις από πολλαπλούς αισθητήρες και ελαχιστοποιώντας τα σφάλματα που οφείλονται σε θόρυβο και αβεβαιότητες. Έτσι, η προσθήκη του έχει την ευκαιρία να βελτιώσει σημαντικά την ακρίβεια των μετρήσεων από τους αισθητήρες UWB και IMU, παρέχοντας πιο αξιόπιστα δεδομένα για την πλοήγηση του drone. Επιπλέον, μπορεί να βοηθήσει στην πρόβλεψη της μελλοντικής κίνησης του drone, επιτρέποντας έτσι τον προγραμματισμό προληπτικών ενεργειών για την αποφυγή εμποδίων και τη διατήρηση της σταθερότητας.

Τελευταία αλλά εξίσου σημαντική είναι η επέκταση η οποία αναφέρεται στην υλοποίηση πειραματικής διάταξης για την δοκιμή του συστήματος σε πραγματικές συνθήκες. Αυτό δύναται να υλοποιηθεί στον χώρο του εργαστηρίου του κτηρίου Ξ της Σχολής των Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου. Η δημιουργία του πειραματικού περιβάλλοντος περιλαμβάνει την κατασκευή θέσεων για την τοποθέτηση των αγκυρών αλλά και την ενσωμάτωση της συσκευής ετικέτας πάνω στο drone. Παράλληλα, κατά την πειραματική διαδικασία ο χειριστής έχει την ικανότητα να ενσωματώσει τον κώδικα τον οποίο θα έχει δημιουργήσει και βελτιώσει στον ελεγχτή του drone μέσω μίας συσκευής USB. Συνεπώς, θα έχει την ικανότητα να προγραμματίσει και να εκτελέσει σενάρια πτήσης με σκοπό να μετατρέψει την προσομοίωση σε εφαρμογή στον πραγματικό κόσμο. Αυτό, είναι πιθανόν, να μπορεί να το υλοποιήσει με συνδυασμό εφαρμογών όπως αυτής του Gazebo με εκείνη του QGroundContol. Με αυτόν τον τρόπο, η πραγματοποίηση πτήσεων σε πραγματικές συνθήκες θα επιτρέψει την αξιολόγηση της απόδοσης του συστήματος υπό διάφορες συνθήκες και την αναγνώριση πιθανών προβλημάτων που δεν εντοπίζονται στις προσομοιώσεις. Επίσης, θα δώσει τη δυνατότητα για τη δοκιμή και την επικύρωση των αλγορίθμων πλοήγησης, της αποφυγής εμποδίων και των τεχνικών προσγείωσης και απογείωσης σε ένα πιο ρεαλιστικό περιβάλλον.

# 8 ΒΙΒΛΙΟΓΡΑΦΙΑ


[1] UNIT 1 - DRONE ELECTRONICS – SECA4003, India: SCHOOL OF ELECTRICAL AND ELECTRONICS-DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING.

[2] Π. Δανηίλ, «Κατασκευή μη επανδρωμένου τετρακόπτερου (Drone),» Πανεπιστήμιο Ιωαννίνων-Τμήμα Πληροφορικής και Τηλεποικοινωνίων, Άρτα, 2019.

[3] S. M. Asaad και H. S. Maghdid, «A Comprehensive Review of Indoor/Outdoor Localization Solutions in IoT,» 2022.

[4] M. Alhafnawi, H. A. B. Salameh, A. Masadeh, H. Al-Obiedollah, M. Ayyash, R. El-Khazali και H. Elgala, «A Survey of Indoor and Outdoor UAV-Based Tracking Systems:Current Status,Challenges, Technologies, and Future Directions,» IEEE, 2023.

[5] V. Bui, N. T. Le, T. L. Vu, V. H. Nguyen και V. M. Jang, «GPS-Based Indoor/Outdoor Detection Scheme Using Machine Learning Techniques,» MDPI, 2020.

[6] P. Savvakis , G. C. Vosniakos, E. Stathatos, A. D. Monclair και M. Chodnichi, «Indoor navigation of a transportation drone in Flexible Manufacturing by integration of open systems».

[7] G. C. Vosniakos, E. Lekai και G. Maltezos, «On the mechanical design of a customized Unmanned Aerial Vehicle transporter for Flexible Manufacturing Systems,» 2022.

[8] Μ. Γεράσιμος, «Μεταφορά υλικών σε Ευέλικτα Συστήματα Κατεργασιών με drone: μελέτη καταλληλότητας πρωτοτύπου,» Εθνικό Μετσόβιο Πολυτεχνείο, Αθήνα, 2020.

[9] JOUAV, «Jouav.com,» 26 June 2024. [Ηλεκτρονικό]. Available: https://www.jouav.com/blog/drone-types.html. [Πρόσβαση 27 June 2024].

[10] E. Tasuns, «Tasuns,» 04 May 2023. [Ηλεκτρονικό]. Available: https://gr.t-composites.net/info/carbon-fiber-in-aerospace-83800979.html. [Πρόσβαση 24 June 2024].

[11] Gernitex, «Gernitex,» 2023. [Ηλεκτρονικό]. Available: https://gernitex.com/resources/carbon-fiber-properties/. [Πρόσβαση 24 June 2024].

[12] E. Tasuns, «Tansuns,» 29 May 2023. [Ηλεκτρονικό]. [Πρόσβαση 24 June 2024].

[13] 3M, «3M™ Scotch-Weld™ Structural Epoxy Adhesive EC-9323 B/A,» 2022.

[14] A. Ademovic, «An Introduction to Robot Operating System: The Ultimate Robot Application Framework,» Toptal Developers, [Ηλεκτρονικό]. Available: https://www.toptal.com/robotics/introduction-to-robot-operating-system. [Πρόσβαση June 2024].

[15] L. B. G. ROBOTS, «ROS- Robot Operating System,» ROS, [Ηλεκτρονικό]. Available: https://www.generationrobots.com/blog/en/ros-robot-operating-system-2/. [Πρόσβαση June 2024].

[16] Gazebo, «Why Gazebo?,» Gazebo, 2014. [Ηλεκτρονικό]. Available: https://classic.gazebosim.org. [Πρόσβαση June 2024].

[17] PX4, «Controller Diagrams,» PXY Guide (main), [Ηλεκτρονικό]. Available: https://docs.px4.io/main/en/flight_stack/controller_diagrams.html. [Πρόσβαση June 2024].

[18] P. Autopilot, «PX4 System Architecture,» PX4 Autopilot, [Ηλεκτρονικό]. Available: https://docs.px4.io/main/en/concept/px4_systems_architecture.html. [Πρόσβαση June 2024].

[19] Akshata, «PX4 vs. ArduPilot: Choosing the Right Open-Source Flight Stack,» The Droning Company, 29 September 2023. [Ηλεκτρονικό]. Available: https://www.thedroningcompany.com/blog/px4-vs-ardupilot-choosing-the-right-open-source-flight-stack. [Πρόσβαση June 2024].

[20] G. Breed, «A Summary of FCC Rules for Ultra Wideband Communications,» High Frequency Electronics, 2005.

[21] G. Scarati, «UAV precise ATOL techniques using UWB technology,» Politecnico di Torino, Torino, 2021.

[22] G. Fantin, «UWB localization system for partially GPS-denied robotic applications,» Collegio di Ingegneria Informatica, del Cinema e Meccatronica Politecnico di Torino, Torino, 2019.

[23] A. Sarkan και S. Sultana, «Study on Ultra-Wideband (UWB) System and Its Applications».

[24] E. Goldoni, A. Savioli, M. Risi και P. Gamba, «Experimental Analysis of RSSI-based Indoor Localization with IEEE 802.15.4,» University of Pavia, Pavia,Italy, 2010.

[25] M. Shchekotov, «Indoor Localization Method Based on Wi-Fi Trilateration Technique,» St.Petersburg, Russia, 2014.

[26] Q. A. a. you, «WHY UWB IS THE PREMIER LOCATION TECHNOLOGY,» Qorno all around you, [Ηλεκτρονικό]. Available: https://www.qorvo.com/innovation/ultra-wideband/technology. [Πρόσβαση June 2024].

[27] Sewio, «Two Way Ranging,» Sewio, [Ηλεκτρονικό]. Available: https://www.sewio.net/uwb-technology/two-way-ranging/. [Πρόσβαση June 2024].

[28] A. Navigation, «Inertial Measurement Unit (IMU) – An Introduction,» Advanced Navigation, 7 June 2024. [Ηλεκτρονικό]. Available: https://www.advancednavigation.com/tech-articles/inertial-measurement-unit-imu-an-introduction/. [Πρόσβαση June 2024].

[29] I. Labs, «Advantages and Disadvantages of Inertial Measurement Units,» Inertial Labs, June 2023. [Ηλεκτρονικό]. Available: https://inertiallabs.com/advantages-and-disadvantages-of-inertial-measurement-units/. [Πρόσβαση June 2024].

# 9 ΠΑΡΑΡΤΗΜΑ

Ο σκοπός του παρατήματος είναι η παράθεση όλων των αλγορίθμων και των κατασκευαστικών σχεδίων που γράφτηκαν κατά την εκτέλεση της διπλωματικής εργασίας.

## 9.1 Ο κώδικας του offb node

```
/**
 * @file offb_node.cpp
 * @brief Offboard control example node, written with MAVROS version 0.19.x, PX4 Pro Flight
 * Stack and tested in Gazebo SITL
 */
// ALll the libraries that are included are here
#include <ros/ros.h>
#include <geometry_msgs/PoseStamped.h>
#include <mavros_msgs/CommandBool.h>
#include <mavros_msgs/SetMode.h>
#include <mavros_msgs/State.h>
#include <sensor_msgs/Imu.h>
#include <geometry_msgs/TwistStamped.h>
#include <geometry_msgs/Vector3.h>
#include <geometry_msgs/Quaternion.h>
#include <tf2_ros/transform_listener.h>
#include <geometry_msgs/TransformStamped.h>
#include <offb/uwbdata.h>
#include <std_msgs/Float32.h>
#include <std_msgs/Float32MultiArray.h>
#include <gazebo_msgs/ModelStates.h>
#include <geometry_msgs/Pose.h>
#include <geometry_msgs/Twist.h>
//#include "position_calculation.h"
#include <vector>
#include <string>
#include <chrono>
#include <thread>
#include <cmath>
#include <Eigen/Dense>
#include <Eigen/Eigenvalues>
#include <iostream>
#include <complex>
#include <algorithm>
#include<iostream>
// end of libraries


//Make the variables global in order for the code to work
mavros_msgs::State current_state;
geometry_msgs:: Vector3 acc_data;
geometry_msgs:: Vector3 vel_data;
geometry_msgs:: Quaternion position_data;
Eigen::MatrixXd anchors(4,3);
 ros::Publisher pos_pub;
// Eigen::Vector3d:: tag_position;
ros::Publisher distance_pub;
geometry_msgs::Point pos_msg;
Eigen ::Vector3d tag_position;
Eigen::VectorXd dist ;
Eigen :: VectorXd robot_pos;
geometry_msgs::Pose locali_pose;
geometry_msgs::Twist locali_twist;
Eigen::Matrix3d rotation_matrix;


// I have to declare everything
```

```
Eigen::Vector3d position_calculation(const Eigen::MatrixXd& anchors, const Eigen::VectorXd& dist);

void subscribeDataCallback(const offb::uwbdata::ConstPtr& msg);

void imuCallback(const sensor_msgs::Imu::ConstPtr& msg);

void state_cb(const mavros_msgs::State::ConstPtr& msg);

double computeDistance(const Eigen:: Vector3d& tag_position, int anchor_index);

Eigen :: VectorXd robotposition(const gazebo_msgs::ModelStates::ConstPtr& msg);

void UWBcallback(const gazebo_msgs::ModelStates::ConstPtr& msg);

Eigen::MatrixXd initializeAnchors();



// Declare the namespaces

using namespace Eigen;

using namespace std;



// Fuction to receive the data that comes from the IMU plugin

void imuCallback(const sensor_msgs::Imu::ConstPtr& msg) {

  acc_data = msg->linear_acceleration; //message for linear acceletation

  vel_data = msg->angular_velocity; // message for angular velocity

  position_data = msg->orientation; // message for orientation


}



// Fuction to receive the data that shows the state of the drone

void state_cb(const mavros_msgs::State::ConstPtr& msg) {

  current_state = *msg;

}



// Fuction to edit the data that comes from the ModelStates and mimic the UWB logic

void UWBcallback(const gazebo_msgs::ModelStates::ConstPtr& msg){

   // locali_pose = msg->pose[index]; Do not need it

  // locali_twist = msg->twist[index]; Do not need it

    Eigen::VectorXd distances = robotposition(msg); // Calculates the distances between the tag and the anchors

    Eigen :: Vector3d robot_pos = position_calculation(anchors,distances); // Calculates the final position of the UAV

    // std::cout << "Robot: " << robot_pos.transpose() << std::endl; MESSAGE FOR DEBUGGING


     // Store the positions

  geometry_msgs::Point pos_msg;

  pos_msg.x = robot_pos[0];

  pos_msg.y = robot_pos[1];

  pos_msg.z = robot_pos[2];

  //Make a publisher to publish the data to a topic

 pos_pub.publish(pos_msg);


}



int main(int argc, char **argv)

{

  //Create and define the node

  ros::init(argc, argv, "offb_node");

  ros::NodeHandle nh;
```

```
// It was a try to initialize the anchors through fuction. It is not used

// Define anchors with their coordinates and dist;

//Eigen::MatrixXd anchors = initializeAnchors();


// Initialize the anchros

// I use four anchors in the same height


//Eigen::MatrixXd anchors(4, 3);

   anchors<<   10.0, 10.0, 3.0,

            10.0, -10.0, 3.0,

            -10.0, 10.0, 3.0 ,

            -10.0, -10.0, 3.0;


// All the publishers and subscribes


   ros::Subscriber state_sub = nh.subscribe<mavros_msgs::State>

         ("mavros/state", 10, state_cb); // Subscribe the state of the drone


   ros::Publisher local_pos_pub = nh.advertise<geometry_msgs::PoseStamped>

         ("mavros/setpoint_position/local", 10); // Publish the local position of the drone


   ros::ServiceClient arming_client = nh.serviceClient<mavros_msgs::CommandBool>

         ("mavros/cmd/arming"); //Show the arming situation


   ros::ServiceClient set_mode_client = nh.serviceClient<mavros_msgs::SetMode>

         ("mavros/set_mode"); //Show the mode of the drone


      // For the IMU system

      // Make a subscriber that collects the information from the IMU plugin

   ros::Subscriber imu_sub = nh.subscribe<sensor_msgs::Imu>("/mavros/imu/data", 10, imuCallback);


   // Make publishers to receive the data and publish them in different topics

   ros::Publisher imu_pub = nh.advertise<geometry_msgs::Vector3>("sofia/imu/linearaccel", 10);

   ros::Publisher imu_pub1 = nh.advertise<geometry_msgs::Vector3>("sofia/imu/velocity", 10);

   ros::Publisher imu_pub2 = nh.advertise<geometry_msgs::Quaternion>("sofia/imu/orientation", 10);


  // For the UWB system. The publisher that prints the position of the drone

   pos_pub = nh.advertise<geometry_msgs::Point>("robot_position", 10);



   // Subscriber for model states

   ros::Subscriber sub = nh.subscribe<gazebo_msgs::ModelStates>("/gazebo/model_states", 10, UWBcallback);



   //The setpoint publishing rate MUST be faster than 2Hz

   ros::Rate rate(20.0);

   ros::Rate loop_rate(20.0);



   // wait for FCU connection

   while(ros::ok() && !current_state.connected){

      ros::spinOnce();

      rate.sleep();

   }

// Senaria in order to try the code
```

```
geometry_msgs::PoseStamped pose;

 // Ensure  neutral orientantion it means no rotation
pose.pose.orientation.x = 0.0;
pose.pose.orientation.y = 0.0;
pose.pose.orientation.z = 0.0;
pose.pose.orientation.w = 1.0;



//Senario1 takeoff
    pose.pose.position.x = 0.0;
   pose.pose.position.y = 0.0;
    pose.pose.position.z = 2.0;

 //Senario2 and 4
 std::vector<std::vector<double>> points = {
   //Senario 2
      {0.0, 0.0, 2.0},
    {1.0, 0.0, 2.0},
    {1.5, 0.0, 2.0},
    {3.0, 0.0, 2.0}
   //Senario 4

 //     {1.0, 1.0, 2.0},
  //    {3.0, 1.0, 2.0},
  //    {3.0, 4.0, 2.0},
  //    {1.0, 4.0, 2.0}
   };
//Senario3
//  pose.pose.position.x = 7.0;
// pose.pose.position.y = 3.0;
 //pose.pose.position.z = 2.0;

  //Senario5
 //  pose.pose.position.x = 11.0;
 //  pose.pose.position.y = 11.0;
 //  pose.pose.position.z = 2.0;

   //send a few setpoints before starting for safety
   for(int i = 100; ros::ok() && i > 0; --i){
      local_pos_pub.publish(pose);
      ros::spinOnce();
      rate.sleep();
   }

 //This code makes the offboard situation and arms the drone

   mavros_msgs::SetMode offb_set_mode;
   offb_set_mode.request.custom_mode = "OFFBOARD";

   mavros_msgs::CommandBool arm_cmd;
   arm_cmd.request.value = true;

   ros::Time last_request = ros::Time::now();

while(ros::ok()){
   if( current_state.mode != "OFFBOARD" &&
      (ros::Time::now() - last_request > ros::Duration(5.0))){
      if( set_mode_client.call(offb_set_mode) &&
         offb_set_mode.response.mode_sent){
         ROS_INFO("Offboard enabled");
      }
      last_request = ros::Time::now();
   } else {
      if( !current_state.armed &&
         (ros::Time::now() - last_request > ros::Duration(5.0))){
         if( arming_client.call(arm_cmd) &&
            arm_cmd.response.success){
            ROS_INFO("Vehicle armed");
```

```
        }
        last_request = ros::Time::now();
      }
    }
  //Senario2
 // Publish setpoints in a sequence to make the drone fly in a straight line
     for (size_t i = 0; i < points.size(); ++i) {
        pose.header.stamp = ros::Time::now();
       pose.pose.position.x = points[i][0];
      pose.pose.position.y = points[i][1];
      pose.pose.position.z = points[i][2];

       for (int j = 0; j < 50; ++j) { // Adjust the inner loop to control how long the drone stays at each point
          if (ros::ok()) {
             local_pos_pub.publish(pose);
              ros::spinOnce();
             rate.sleep();
           }
       }
     }

  local_pos_pub.publish(pose);

  ros::spinOnce();
  rate.sleep();

  // Publish all the imu data

  imu_pub.publish(acc_data);
  imu_pub1.publish(vel_data);
  imu_pub2.publish(position_data);

   // These commands make the publishers run
  ros::spinOnce();
  loop_rate.sleep();

// ros::spinOnce();
//loop_rate.sleep();
}
return 0;
}

// For the anchors. It was a try to initialize the anchors through a fuction. The fuction is not in use
//Eigen::MatrixXd initializeAnchors() {
  // Eigen::MatrixXd anchors(4, 3);
   //anchors << 10.0, 10.0, 3.0,
        //    10.0, -10.0, 3.0,
       //     -10.0, 10.0, 3.0,
      //      -10.0, -10.0, 3.0;
 //   return anchors;
//}

// Function to calculate distance with the square least localization logic
 Eigen :: Vector3d position_calculation(const Eigen::MatrixXd& anchors, const Eigen::VectorXd& dist) {

  Eigen :: MatrixXd A = (-2 * anchors);
   int vertical = anchors.cols();
   int horizontal = anchors.rows();
   //For Debugging
```

```
  //   std::cout << "Anchors: \n" << anchors << std::endl;

  // std::cout << "Dist: \n" << dist << std::endl;

  //   ROS_INFO("Anchors (rows, cols): (%ld, %ld)", anchors.rows(), anchors.cols());

  // ROS_INFO("Dist size: %ld", dist.size());


   Eigen :: MatrixXd B = Eigen :: MatrixXd::Ones(horizontal, 1);

  //   A.conservativeResize(Eigen ::NoChange, A.cols()+1);

   // You have to make the table to have for columns with the fist one to have zeros

    Eigen:: MatrixXd Anew(4,4);

    Anew.col(0)=B;

    Anew.col(1) = A.col(0);

    Anew.col(2) = A.col(1);

    Anew.col(3) =A.col(2);


// For Debugging

  //ROS_INFO("A (rows, cols): (%ld, %ld)", A.rows(), A.cols());

//std::cout << "A: \n" << Anew << std::endl;

   Eigen :: VectorXd R = Eigen::VectorXd::Zero(vertical); // I do not use it

   Eigen :: MatrixXd b = Eigen::MatrixXd::Zero(horizontal,1);

   //I make the b matrix the way theory shows

  for (int i = 0; i < horizontal; ++i) {


     b(i,0) = (dist(i)*dist(i)) - (anchors(i,0)*anchors(i,0))- (anchors(i,1)*anchors(i,1))- (anchors(i,2)*anchors(i,2));


    }


      // Debugging message

  // ROS_INFO("b size: %ld", b.size());


   Eigen:: MatrixXd Atr = Anew.transpose(); // Find the A^T

   Eigen::MatrixXd AtrA(4, 4); // Declare a new matrix


   // Multiply the tables. It was not okay to just use the * sumbol

     for (int i = 0; i < Atr.rows(); ++i) {

       for (int j = 0; j < Anew.cols(); ++j) {

          AtrA(i, j) = 0; // Initialize result element at (i, j)


          for (int k = 0; k < Atr.cols(); ++k) {

             AtrA(i, j) += Atr(i, k) * Anew(k, j);

          }

       }

    }


// For debugging. It is important to know if the table is inverible


   //Eigen:: MatrixXd AtrA =Atr* A;

   // Check if AtrA is invertible

    //if (AtrA.determinant() == 0) {

    //    ROS_ERROR("Matrix AtrA is not invertible!");

    //    return Eigen::Vector3d::Zero();

  //  }


   // std::cout << "Matrix A:\n" << A << std::endl;

   // std::cout << "Matrix A^T:\n" << Atr << std::endl;

   //  std::cout << "Matrix A^T * A:\n" << AtrA << std::endl;

    //std::cout << "Determinant of A^T * A: " << det << std::endl;


   // Do the inverse


Eigen::CompleteOrthogonalDecomposition<Eigen:: MatrixXd> codA(A);
```

```
    //Find the pseudoinverse

    Eigen::MatrixXd pseudoAtrA = codA.pseudoInverse();



    //Use of debugging messages

    // std::cout << "Matrix pseudo:\n" << pseudoAtrA << std::endl;

   // Eigen:: MatrixXd Atotal = pseudoAtrA *Anew.transpose();



    Eigen:: MatrixXd yhat = pseudoAtrA*b;

    //Again a debugging message

  // std::cout << "Matrix YHAT:\n" << yhat << std::endl;



  Eigen::Vector3d x = yhat.block<3, 1>(yhat.size() - 3, 0);  // Extract last 3 elements

 //std:: cout << "x:\n"<< x << std:: endl;

 // I want positive values

     if (x[0] <0){

     x[0] = -x[0];


   }

   if(x[1]<0) {

     x[1]=-x[1];

   }


  // Try to debug

//std:: cout << "x:\n"<< x << std:: endl;


   return x;


}



//Fuction compute the distance between the tag and the anchors using ToF logic in order to mimic the UWB

double computeDistance(const Eigen:: Vector3d& tag_position, int anchor_index) {

  Eigen:: Vector3d anchor_position = anchors.row(anchor_index);

   return (tag_position - anchor_position).norm();

}



//Function to caluclate distance between anchros and tags.

//I have to know tag position. The tag position is the same with the drone

Eigen :: VectorXd robotposition(const gazebo_msgs::ModelStates::ConstPtr& msg) {


Eigen:: VectorXd  dist(anchors.rows());

 dist.setZero(); // Initialize with zeros

   // Find the index of the tag in the model states

   std::vector<std::string>::const_iterator it = std::find(msg->name.begin(), msg->name.end(), "iris");

   if (it != msg->name.end()) {

      int tag_index = std::distance(msg->name.begin(), it);

      // Extract tag position

      Eigen ::Vector3d tag_position(msg->pose[tag_index].position.x, msg->pose[tag_index].position.y, msg->pose[tag_index].position.z);

      // Use of debugging message

    // std::cout << "Tag Position: " << tag_position.transpose() << std::endl;


      // Compute distance from each anchor

      for (int i = 0; i < anchors.rows(); ++i) {

        double distance = computeDistance(tag_position, i);

        dist(i) = distance;

        //Use of debugging message
```

```
    // std::cout << "Distance to anchor " << i << ": " << distance << std::endl;
        } }else {
        std::cerr << "'iris' not found in model states." << std::endl;
    }
    return dist;
    }
```

## 9.2 Το αρχείο CMakeList.txt

```
cmake_minimum_required(VERSION 3.0.2)

project(offb)


## Compile as C++11, supported in ROS Kinetic and newer
 add_compile_options(-std=c++11)


## Find catkin macros and libraries
## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  sensor_msgs
  roscpp
  rospy
  mavros_msgs
  geometry_msgs
  gazebo_ros
  std_msgs
  message_generation
  gazebo_msgs
  offb
  tf2_ros
)
set(EIGEN3_INCLUDE_DIR "${CMAKE_SOURCE_DIR}/src/offb/src/Eigen")
find_package(Eigen3 REQUIRED)
## System dependencies are found with CMake's conventions
# find_package(Boost REQUIRED COMPONENTS system)


## Generate messages in the 'msg' folder
 add_message_files(
 FILES
uwbdata.msg


 )


####################################
## catkin specific configuration ##
####################################
## The catkin_package macro generates cmake config files for your package
## Declare things to be passed to dependent projects
## INCLUDE_DIRS: uncomment this if your package contains header files
## LIBRARIES: libraries you create in this project that dependent projects also need
## CATKIN_DEPENDS: catkin_packages dependent projects also need
## DEPENDS: system dependencies of this project that dependent projects also need
catkin_package(
  CATKIN_DEPENDS roscpp sensor_msgs mavros_msgs geometry_msgs gazebo_ros
    message_runtime  std_msgs gazebo_msgs tf2_ros
    DEPENDS Eigen3
)
```

```
############
## Build ##
############

## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
 include
 ${catkin_INCLUDE_DIRS}
 src/offb/src
 ${GAZEBO_INCLUDE_DIRS}
 ${EIGEN3_INCLUDE_DIR}
 ${CMAKE_SOURCE_DIR}/src/offb/src/Eigen
)

## Add cmake target dependencies of the library
## as an example, code may need to be generated before libraries
## either from message generation or dynamic reconfigure
# add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
# add_executable(${PROJECT_NAME}_node src/offb_node.cpp)

##add_executable(${PROJECT_NAME}_node src/offb_node.cpp BMI085_Gyroscope.cpp)
add_executable(${PROJECT_NAME}_node
   src/offb_node.cpp          # existing source file

)

target_link_libraries(${PROJECT_NAME}_node
   ${catkin_LIBRARIES}
   ${YOUR_EXTERNAL_LIBRARIES}

)
add_dependencies(${PROJECT_NAME}_node
   ${catkin_EXPORTED_TARGETS}
   sensor_msgs_generate_messages_cpp

)

target_link_libraries(${PROJECT_NAME}_node
 ${catkin_LIBRARIES}
 ${Eigen3_LIBRARIES}
 )
```

## 9.3 Το αρχείο main.launch

```
<launch>

        <!-- Include the MAVROS node with SITL and Gazebo -->
        <include file="$(find px4)/launch/mavros_posix_sitl.launch">

        </include>

        <!-- Our node to control the drone -->
```

```
        <node pkg="offb" type="offb_node" name="offb_node" required="true" output="screen" />
        <param name="loop_rate" value="400" />
         <param name="use_sim_time" value="true" />
```

```
</launch>
```

## 9.4 Το αρχείο iris.sdf

```
<sdf version='1.6'>
  <model name='iris'>
    <link name='base_link'>
      <pose>0 0 0 0 0 0</pose>
      <inertial>
        <pose>0 0 0 0 0 0</pose>
        <mass>1.5</mass>
        <inertia>
          <ixx>0.029125</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.029125</iyy>
          <iyz>0</iyz>
          <izz>0.055225</izz>
        </inertia>
      </inertial>
      <collision name='base_link_inertia_collision'>
        <pose>0 0 0 0 0 0</pose>
        <geometry>
          <box>
            <size>0.47 0.47 0.11</size>
          </box>
        </geometry>
        <surface>
          <contact>
            <ode>
              <min_depth>0.001</min_depth>
              <max_vel>0</max_vel>
            </ode>
          </contact>
          <friction>
            <ode/>
          </friction>
        </surface>
      </collision>
      <visual name='base_link_inertia_visual'>
        <pose>0 0 0 0 0 0</pose>
        <geometry>
          <mesh>
            <scale>1 1 1</scale>
            <uri>model://iris/meshes/iris.stl</uri>
          </mesh>
        </geometry>
        <material>
          <script>
            <name>Gazebo/DarkGrey</name>
<uri>file://media/materials/scripts/gazebo.material
</uri>
          </script>
        </material>
      </visual>
      <gravity>1</gravity>
      <velocity_decay/>
    </link>
    <link name='/imu_link'>
      <pose>0 0 0.02 0 0 0</pose>
```

```
    <inertial>
     <pose>0 0 0 0 0 0</pose>
     <mass>0.015</mass>
     <inertia>
      <ixx>1e-05</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>1e-05</iyy>
      <iyz>0</iyz>
      <izz>1e-05</izz>
     </inertia>
    </inertial>
   </link>
   <joint name='/imu_joint' type='revolute'>
    <child>/imu_link</child>
    <parent>base_link</parent>
    <axis>
     <xyz>1 0 0</xyz>
     <limit>
      <lower>0</lower>
      <upper>0</upper>
      <effort>0</effort>
      <velocity>0</velocity>
     </limit>
     <dynamics>
      <spring_reference>0</spring_reference>
      <spring_stiffness>0</spring_stiffness>
     </dynamics>

<use_parent_model_frame>1</use_parent_model
_frame>
    </axis>
   </joint>
   <link name='rotor_0'>
    <pose>0.13 -0.22 0.023 0 0 0</pose>
    <inertial>
     <pose>0 0 0 0 0 0</pose>
     <mass>0.005</mass>
     <inertia>
      <ixx>9.75e-07</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.000273104</iyy>
      <iyz>0</iyz>
      <izz>0.000274004</izz>
     </inertia>
    </inertial>
    <collision name='rotor_0_collision'>
     <pose>0 0 0 0 0 0</pose>
     <geometry>
      <cylinder>
       <length>0.005</length>
       <radius>0.128</radius>
      </cylinder>
     </geometry>
     <surface>
      <contact>
       <ode/>
      </contact>
      <friction>
       <ode/>
      </friction>
     </surface>
    </collision>
    <visual name='rotor_0_visual'>
     <pose>0 0 0 0 0 0</pose>
     <geometry>
      <mesh>
       <scale>1 1 1</scale>

<uri>model://iris/meshes/iris_prop_ccw.dae</uri>
      </mesh>
```

```
      </geometry>
      <material>
       <script>
        <name>Gazebo/Blue</name>
<uri>file://media/materials/scripts/gazebo.material
</uri>
       </script>
      </material>
     </visual>
     <gravity>1</gravity>
     <velocity_decay/>
    </link>
    <joint name='rotor_0_joint' type='revolute'>
     <child>rotor_0</child>
     <parent>base_link</parent>
     <axis>
      <xyz>0 0 1</xyz>
      <limit>
       <lower>-1e+16</lower>
       <upper>1e+16</upper>
      </limit>
      <dynamics>
       <spring_reference>0</spring_reference>
       <spring_stiffness>0</spring_stiffness>
      </dynamics>
<use_parent_model_frame>1</use_parent_model
_frame>
     </axis>
    </joint>
    <link name='rotor_1'>
     <pose>-0.13 0.2 0.023 0 0 0</pose>
     <inertial>
      <pose>0 0 0 0 0 0</pose>
      <mass>0.005</mass>
      <inertia>
       <ixx>9.75e-07</ixx>
       <ixy>0</ixy>
       <ixz>0</ixz>
       <iyy>0.000273104</iyy>
       <iyz>0</iyz>
       <izz>0.000274004</izz>
      </inertia>
     </inertial>
     <collision name='rotor_1_collision'>
      <pose>0 0 0 0 0 0</pose>
      <geometry>
       <cylinder>
        <length>0.005</length>
        <radius>0.128</radius>
       </cylinder>
      </geometry>
      <surface>
       <contact>
        <ode/>
       </contact>
       <friction>
        <ode/>
       </friction>
      </surface>
     </collision>
     <visual name='rotor_1_visual'>
      <pose>0 0 0 0 0 0</pose>
      <geometry>
       <mesh>
        <scale>1 1 1</scale>
<uri>model://iris/meshes/iris_prop_ccw.dae</uri>
       </mesh>
      </geometry>
      <material>
       <script>
```

```
          <name>Gazebo/DarkGrey</name>

<uri>file://media/materials/scripts/gazebo.material
</uri>
        </script>
       </material>
     </visual>
     <gravity>1</gravity>
     <velocity_decay/>
    </link>
    <joint name='rotor_1_joint' type='revolute'>
     <child>rotor_1</child>
     <parent>base_link</parent>
     <axis>
      <xyz>0 0 1</xyz>
      <limit>
       <lower>-1e+16</lower>
       <upper>1e+16</upper>
      </limit>
      <dynamics>
       <spring_reference>0</spring_reference>
       <spring_stiffness>0</spring_stiffness>
      </dynamics>

<use_parent_model_frame>1</use_parent_model
_frame>
     </axis>
    </joint>
    <link name='rotor_2'>
     <pose>0.13 0.22 0.023 0 0 0</pose>
     <inertial>
      <pose>0 0 0 0 0 0</pose>
      <mass>0.005</mass>
      <inertia>
       <ixx>9.75e-07</ixx>
       <ixy>0</ixy>
       <ixz>0</ixz>
       <iyy>0.000273104</iyy>
       <iyz>0</iyz>
       <izz>0.000274004</izz>
      </inertia>
     </inertial>
     <collision name='rotor_2_collision'>
      <pose>0 0 0 0 0 0</pose>
      <geometry>
       <cylinder>
        <length>0.005</length>
        <radius>0.128</radius>
       </cylinder>
      </geometry>
      <surface>
       <contact>
        <ode/>
       </contact>
       <friction>
        <ode/>
       </friction>
      </surface>
     </collision>
     <visual name='rotor_2_visual'>
      <pose>0 0 0 0 0 0</pose>
      <geometry>
       <mesh>
        <scale>1 1 1</scale>

<uri>model://iris/meshes/iris_prop_cw.dae</uri>
       </mesh>
      </geometry>
      <material>
       <script>
        <name>Gazebo/Blue</name>

<uri>file://media/materials/scripts/gazebo.material
</uri>
```

```
      </script>
     </material>
    </visual>
    <gravity>1</gravity>
    <velocity_decay/>
   </link>
   <joint name='rotor_2_joint' type='revolute'>
    <child>rotor_2</child>
    <parent>base_link</parent>
    <axis>
     <xyz>0 0 1</xyz>
     <limit>
      <lower>-1e+16</lower>
      <upper>1e+16</upper>
     </limit>
     <dynamics>
      <spring_reference>0</spring_reference>
      <spring_stiffness>0</spring_stiffness>
     </dynamics>

<use_parent_model_frame>1</use_parent_model
_frame>
    </axis>
   </joint>
   <link name='rotor_3'>
    <pose>-0.13 -0.2 0.023 0 0 0</pose>
    <inertial>
     <pose>0 0 0 0 0 0</pose>
     <mass>0.005</mass>
     <inertia>
      <ixx>9.75e-07</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.000273104</iyy>
      <iyz>0</iyz>
      <izz>0.000274004</izz>
     </inertia>
    </inertial>
    <collision name='rotor_3_collision'>
     <pose>0 0 0 0 0 0</pose>
     <geometry>
      <cylinder>
       <length>0.005</length>
       <radius>0.128</radius>
      </cylinder>
     </geometry>
     <surface>
      <contact>
       <ode/>
      </contact>
      <friction>
       <ode/>
      </friction>
     </surface>
    </collision>
    <visual name='rotor_3_visual'>
     <pose>0 0 0 0 0 0</pose>
     <geometry>
      <mesh>
       <scale>1 1 1</scale>

<uri>model://iris/meshes/iris_prop_cw.dae</uri>
      </mesh>
     </geometry>
     <material>
      <script>
       <name>Gazebo/DarkGrey</name>

<uri>file://media/materials/scripts/gazebo.material
</uri>
      </script>
     </material>
    </visual>
```

```
    <gravity>1</gravity>
    <velocity_decay/>
   </link>
   <joint name='rotor_3_joint' type='revolute'>
    <child>rotor_3</child>
    <parent>base_link</parent>
    <axis>
     <xyz>0 0 1</xyz>
     <limit>
      <lower>-1e+16</lower>
      <upper>1e+16</upper>
     </limit>
     <dynamics>
      <spring_reference>0</spring_reference>
      <spring_stiffness>0</spring_stiffness>
     </dynamics>

<use_parent_model_frame>1</use_parent_model
_frame>

    </axis>
   </joint>
   <model name="uwb_anchor_1">
  <static>true</static>
  <link name="link_1">
   <pose>10 10 3 0 0 0</pose>
   <visual name="visual_1">
    <geometry>
     <box>
      <size>0.1 0.1 0.1</size>
     </box>
    </geometry>
    <material>
     <script>

<uri>file://media/materials/scripts/gazebo.materi
al</uri>

      <name>Gazebo/Red</name>
     </script>
    </material>
   </visual>
  </link>
</model>

<model name="uwb_anchor_2">
 <static>true</static>
 <link name="link_2">
  <pose>-10 10 3 0 0 0</pose>
  <visual name="visual_2">
   <geometry>
    <box>
     <size>0.1 0.1 0.1</size>
    </box>
   </geometry>
   <material>
    <script>

<uri>file://media/materials/scripts/gazebo.materi
al</uri>

      <name>Gazebo/Green</name>
     </script>
    </material>
   </visual>
  </link>
</model>

<model name="uwb_anchor_3">
 <static>true</static>
 <link name="link_3">
  <pose>-10 -10 3 0 0 0</pose>
  <visual name="visual_3">
   <geometry>
    <box>
     <size>0.1 0.1 0.1</size>
```

```
      </box>
     </geometry>
     <material>
      <script>

<uri>file://media/materials/scripts/gazebo.material</uri>
       <name>Gazebo/Blue</name>
      </script>
     </material>
    </visual>
   </link>
</model>

<model name="uwb_anchor_4">
  <static>true</static>
  <link name="link_4">
    <pose>10 -10 3 0 0 0</pose>
    <visual name="visual_4">
     <geometry>
      <box>
       <size>0.2 0.2 0.2</size>
      </box>
     </geometry>
     <material>
      <script>

<uri>file://media/materials/scripts/gazebo.material</uri>
       <name>Gazebo/Yellow</name>
      </script>
     </material>
    </visual>
   </link>
</model>

    <plugin name='rosbag' filename='libgazebo_multirotor_base_plugin.so'>
      <robotNamespace/>
      <linkName>base_link</linkName>

<rotorVelocitySlowdownSim>10</rotorVelocitySlowdownSim>
    </plugin>
    <plugin name='front_right_motor_model' filename='libgazebo_motor_model.so'>
      <robotNamespace/>
      <jointName>rotor_0_joint</jointName>
      <linkName>rotor_0</linkName>
      <turningDirection>ccw</turningDirection>
      <timeConstantUp>0.0125</timeConstantUp>

<timeConstantDown>0.025</timeConstantDown>
      <maxRotVelocity>1100</maxRotVelocity>
      <motorConstant>5.84e-06</motorConstant>
      <momentConstant>0.06</momentConstant>

<commandSubTopic>/gazebo/command/motor_speed</commandSubTopic>
      <motorNumber>0</motorNumber>

<rotorDragCoefficient>0.000175</rotorDragCoefficient>
      <rollingMomentCoefficient>1e-06</rollingMomentCoefficient>

<motorSpeedPubTopic>/motor_speed/0</motorSpeedPubTopic>

<rotorVelocitySlowdownSim>10</rotorVelocitySlowdownSim>
    </plugin>
    <plugin name='back_left_motor_model' filename='libgazebo_motor_model.so'>
      <robotNamespace/>
      <jointName>rotor_1_joint</jointName>
      <linkName>rotor_1</linkName>
      <turningDirection>ccw</turningDirection>
      <timeConstantUp>0.0125</timeConstantUp>
```

<timeConstantDown>0.025</timeConstantDown>

<maxRotVelocity>1100</maxRotVelocity>

<motorConstant>5.84e-06</motorConstant>

<momentConstant>0.06</momentConstant>

<commandSubTopic>/gazebo/command/motor_speed</commandSubTopic>

<motorNumber>1</motorNumber>

<rotorDragCoefficient>0.000175</rotorDragCoefficient>

<rollingMomentCoefficient>1e-06</rollingMomentCoefficient>

<motorSpeedPubTopic>/motor_speed/1</motorSpeedPubTopic>

<rotorVelocitySlowdownSim>10</rotorVelocitySlowdownSim>

</plugin>

<plugin name='front_left_motor_model' filename='libgazebo_motor_model.so'>

<robotNamespace/>

<jointName>rotor_2_joint</jointName>

<linkName>rotor_2</linkName>

<turningDirection>cw</turningDirection>

<timeConstantUp>0.0125</timeConstantUp>

<timeConstantDown>0.025</timeConstantDown>

<maxRotVelocity>1100</maxRotVelocity>

<motorConstant>5.84e-06</motorConstant>

<momentConstant>0.06</momentConstant>

<commandSubTopic>/gazebo/command/motor_speed</commandSubTopic>

<motorNumber>2</motorNumber>

<rotorDragCoefficient>0.000175</rotorDragCoefficient>

<rollingMomentCoefficient>1e-06</rollingMomentCoefficient>

<motorSpeedPubTopic>/motor_speed/2</motorSpeedPubTopic>

<rotorVelocitySlowdownSim>10</rotorVelocitySlowdownSim>

</plugin>

<plugin name='back_right_motor_model' filename='libgazebo_motor_model.so'>

<robotNamespace/>

<jointName>rotor_3_joint</jointName>

<linkName>rotor_3</linkName>

<turningDirection>cw</turningDirection>

<timeConstantUp>0.0125</timeConstantUp>

<timeConstantDown>0.025</timeConstantDown>

<maxRotVelocity>1100</maxRotVelocity>

<motorConstant>5.84e-06</motorConstant>

<momentConstant>0.06</momentConstant>

<commandSubTopic>/gazebo/command/motor_speed</commandSubTopic>

<motorNumber>3</motorNumber>

<rotorDragCoefficient>0.000175</rotorDragCoefficient>

<rollingMomentCoefficient>1e-06</rollingMomentCoefficient>

<motorSpeedPubTopic>/motor_speed/3</motorSpeedPubTopic>

<rotorVelocitySlowdownSim>10</rotorVelocitySlowdownSim>

</plugin>

<include>

<uri>model://gps</uri>

<pose>0.05 0 0.04 0 0 0</pose>

<name>gps0</name>

</include>

<joint name='gps0_joint' type='fixed'>

<child>gps0::link</child>

```
    <parent>base_link</parent>
  </joint>
  <plugin name='groundtruth_plugin' filename='libgazebo_groundtruth_plugin.so'>
    <robotNamespace/>
  </plugin>
  <plugin name='magnetometer_plugin' filename='libgazebo_magnetometer_plugin.so'>
    <robotNamespace/>
    <pubRate>100</pubRate>
    <noiseDensity>0.0004</noiseDensity>
    <randomWalk>6.4e-06</randomWalk>
    <biasCorrelationTime>600</biasCorrelationTime>
    <magTopic>/mag</magTopic>
  </plugin>
  <plugin name='barometer_plugin' filename='libgazebo_barometer_plugin.so'>
    <robotNamespace/>
    <pubRate>50</pubRate>
    <baroTopic>/baro</baroTopic>
    <baroDriftPaPerSec>0</baroDriftPaPerSec>
  </plugin>
  <plugin name='mavlink_interface' filename='libgazebo_mavlink_interface.so'>
    <robotNamespace/>
    <imuSubTopic>/imu</imuSubTopic>
    <magSubTopic>/mag</magSubTopic>
    <baroSubTopic>/baro</baroSubTopic>
    <mavlink_addr>INADDR_ANY</mavlink_addr>
    <mavlink_tcp_port>4560</mavlink_tcp_port>
    <mavlink_udp_port>14560</mavlink_udp_port>
    <serialEnabled>0</serialEnabled>
    <serialDevice>/dev/ttyACM0</serialDevice>
    <baudRate>921600</baudRate>
    <qgc_addr>INADDR_ANY</qgc_addr>
    <qgc_udp_port>14550</qgc_udp_port>
    <sdk_addr>INADDR_ANY</sdk_addr>
    <sdk_udp_port>14540</sdk_udp_port>
    <hil_mode>0</hil_mode>
    <hil_state_level>0</hil_state_level>
    <send_vision_estimation>0</send_vision_estimation>
    <send_odometry>1</send_odometry>
    <enable_lockstep>1</enable_lockstep>
    <use_tcp>1</use_tcp>
    <motorSpeedCommandPubTopic>/gazebo/command/motor_speed</motorSpeedCommandPubTopic>
    <control_channels>
      <channel name='rotor1'>
        <input_index>0</input_index>
        <input_offset>0</input_offset>
        <input_scaling>1000</input_scaling>
        <zero_position_disarmed>0</zero_position_disarmed>
        <zero_position_armed>100</zero_position_armed>
        <joint_control_type>velocity</joint_control_type>
      </channel>
      <channel name='rotor2'>
        <input_index>1</input_index>
        <input_offset>0</input_offset>
        <input_scaling>1000</input_scaling>
        <zero_position_disarmed>0</zero_position_disarmed>
        <zero_position_armed>100</zero_position_armed>
        <joint_control_type>velocity</joint_control_type>
      </channel>
      <channel name='rotor3'>
```

```
        <input_index>2</input_index>
        <input_offset>0</input_offset>
        <input_scaling>1000</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>

<zero_position_armed>100</zero_position_armed>

<joint_control_type>velocity</joint_control_type>
      </channel>
      <channel name='rotor4'>
        <input_index>3</input_index>
        <input_offset>0</input_offset>
        <input_scaling>1000</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>

<zero_position_armed>100</zero_position_armed>

<joint_control_type>velocity</joint_control_type>
      </channel>
      <channel name='rotor5'>
        <input_index>4</input_index>
        <input_offset>1</input_offset>
        <input_scaling>324.6</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>

<zero_position_armed>0</zero_position_armed>

<joint_control_type>velocity</joint_control_type>
        <joint_control_pid>
          <p>0.1</p>
          <i>0</i>
          <d>0</d>
          <iMax>0.0</iMax>
          <iMin>0.0</iMin>
          <cmdMax>2</cmdMax>
          <cmdMin>-2</cmdMin>
        </joint_control_pid>

<joint_name>zephyr_delta_wing::propeller_joint</joint_name>
      </channel>
      <channel name='rotor6'>
        <input_index>5</input_index>
        <input_offset>0</input_offset>
        <input_scaling>0.524</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>

<zero_position_armed>0</zero_position_armed>

<joint_control_type>position</joint_control_type>

<joint_name>zephyr_delta_wing::flap_left_joint</joint_name>
        <joint_control_pid>
          <p>10.0</p>
          <i>0</i>
          <d>0</d>
          <iMax>0</iMax>
          <iMin>0</iMin>
          <cmdMax>20</cmdMax>
          <cmdMin>-20</cmdMin>
        </joint_control_pid>
      </channel>
      <channel name='rotor7'>
        <input_index>6</input_index>
        <input_offset>0</input_offset>
        <input_scaling>0.524</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>
```

<zero_position_armed>0</zero_position_armed>

<joint_control_type>position</joint_control_type>

<joint_name>zephyr_delta_wing::flap_right_joint</joint_name>

<joint_control_pid>

10.0

0

<d>0</d>

<iMax>0</iMax>

<iMin>0</iMin>

<cmdMax>20</cmdMax>

<cmdMin>-20</cmdMin>

</joint_control_pid>

</channel>

<channel name='rotor8'>

<input_index>7</input_index>

<input_offset>0</input_offset>

<input_scaling>0.524</input_scaling>

<zero_position_disarmed>0</zero_position_disarmed>

<zero_position_armed>0</zero_position_armed>

<joint_control_type>position</joint_control_type>

</channel>

</control_channels>

</plugin>

<static>0</static>

**<plugin name='rotors_gazebo_imu_plugin' filename='libgazebo_imu_plugin.so'>**

**<robotNamespace/>**

**<linkName>/imu_link</linkName>**

**<frameId>base_link_link</frameId>**

**<alwaysOn>true</alwaysOn>**

**<updateRate>400.0</updateRate>**

**<imuTopic>/imu</imuTopic>**

<gyroscopeNoiseDensity>0.00018665</gyroscopeNoiseDensity>

<gyroscopeRandomWalk>3.8785e-05</gyroscopeRandomWalk>

<gyroscopeBiasCorrelationTime>1000.0</gyroscopeBiasCorrelationTime>

<gyroscopeTurnOnBiasSigma>0.0087</gyroscopeTurnOnBiasSigma>

<accelerometerNoiseDensity>0.00186</accelerometerNoiseDensity>

<accelerometerRandomWalk>0.006</accelerometerRandomWalk>

<accelerometerBiasCorrelationTime>300.0</accelerometerBiasCorrelationTime>

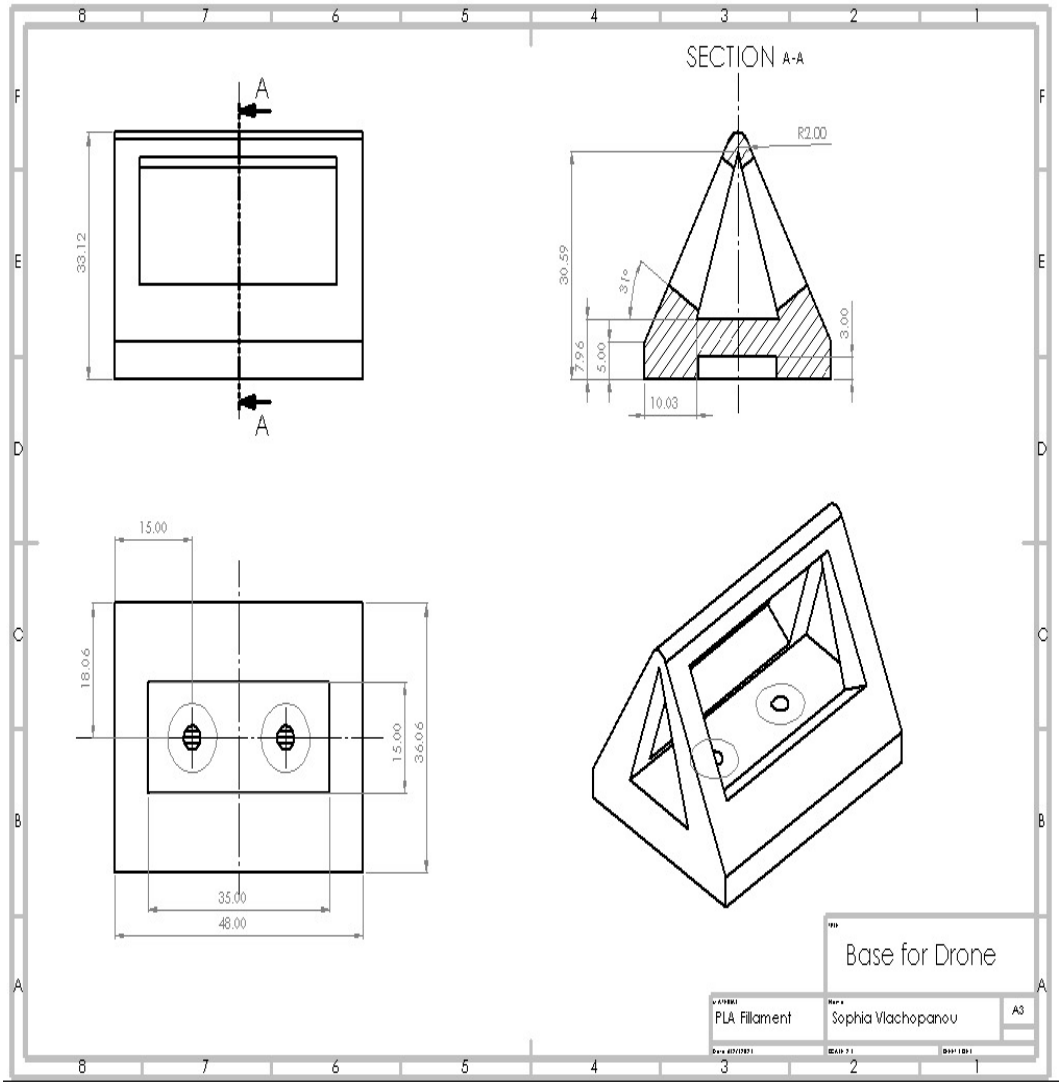
<accelerometerTurnOnBiasSigma>0.196</accelerometerTurnOnBiasSigma>

</plugin>

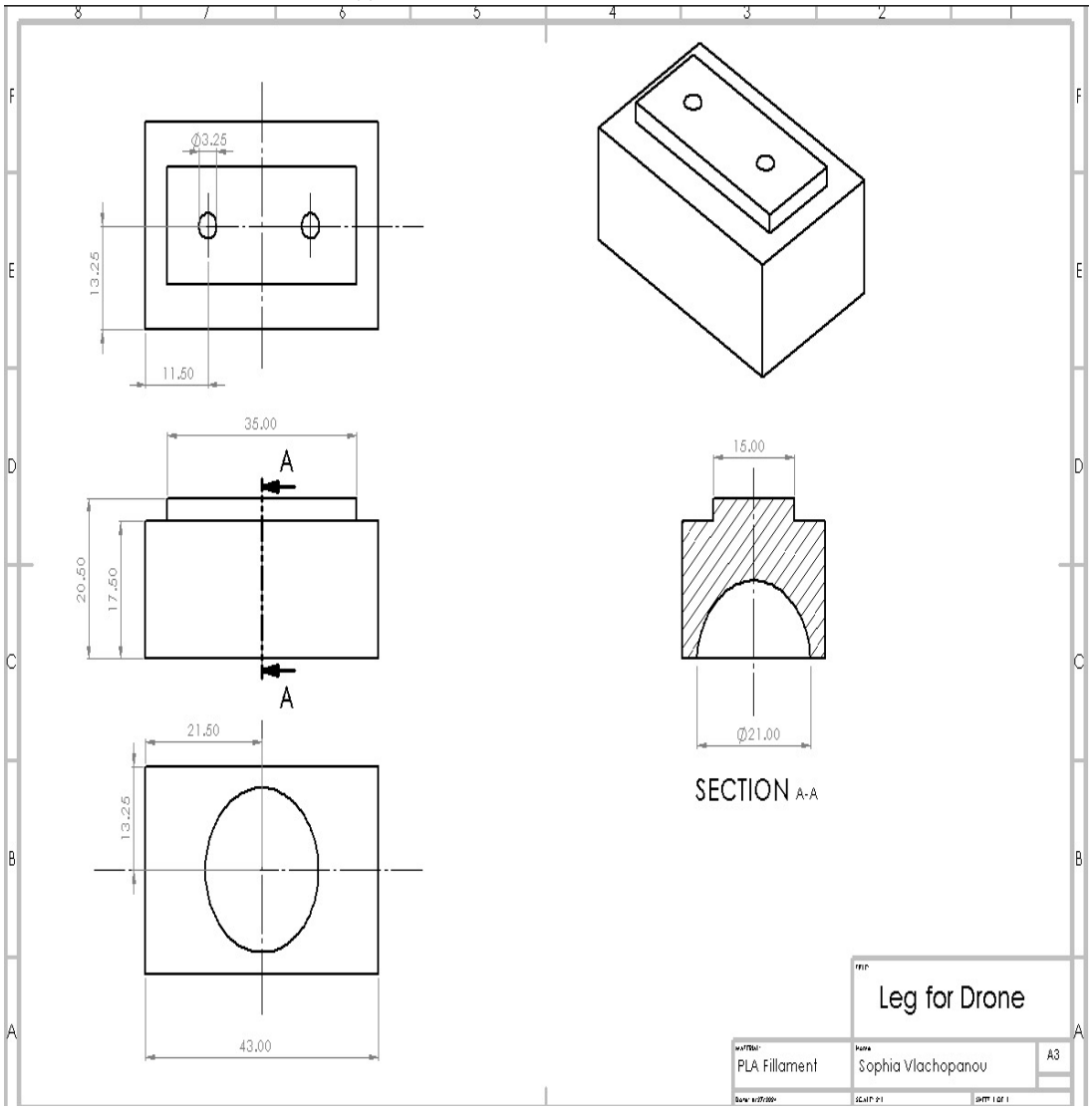
</model>

</sdf>

## 9.5 Κατασκευαστικό σχέδιο βάσης τοποθέτησης ποδιού



## 9.6 Κατασκευαστικό σχέδιο ποδιού drone



## 9.7 Κατασκευαστικό σχέδιο του εξαρτήματος για την βάση